Aplicación de la Búsqueda Local Iterada a la solución del problema de ruteo de vehículos con múltiples depósitos y flota propia y subcontratada MDVRPPC.

JUAN MANUEL AMARILES ZAMBRANO

Facultad de Ingeniería Industrial
Maestría en Investigación Operativa y Estadística
Universidad Tecnológica de Pereira
Pereira, Risaralda
2019

1 Contenido

2 Índice de Algoritmos

3 Índice de Figuras

4 Índice de Tablas

5 Información General

Título: Aplicación de la Búsqueda Local Iterada a la solución de ruteo de vehículos con múltiples depósitos y flota propia y subcontratada MDVRPPC.

Área de investigación: Esta propuesta de investigación se enmarca en la línea de Optimización aproximada, por ser una propuesta que aporta una aplicación de los modelos cuantitativos, a problemas generados en los sistemas de transporte terrestre.

Materias asociadas a la investigación: Programación lineal entera mixta, Técnicas heurísticas y Metaheurísticas.

Director: Eliana Mirledy Toro Ocampo. Ing. Industrial, Doctora en Ingeniería, Magíster en Ingeniería Eléctrica en la línea de Investigación operativa y Magíster en Investigación de Operaciones y Estadística. Profesora Titular, Facultad de ingeniería industrial. Universidad Tecnológica de Pereira. E-mail: elianam@utp.edu.co

6 Planteamiento del Problema

Desde que se planteó por primera vez en los años cincuenta el problema de ruteo de vehículos (VRP) ha sido ampliamente tratado dada su aplicabilidad en múltiples escenarios de carácter logístico. Puesto que en la actualidad la economía de una región está directamente relacionada con la eficiencia con la que operan sus propias fuentes económicas, el uso de este tipo de técnicas matemáticas es cada vez más frecuente. En el campo de la distribución y la recogida de productos, por ejemplo, el costo final que pueda tener dicho producto está relacionado con la manera en que se gestione el proceso de entrega de estos. Dada la gran cantidad de factores relacionados con la planeación de las rutas de distribución, el problema de ruteo de vehículos constituye un área fundamental en la gestión de las empresas dedicadas al transporte.

Una situación particular que se presenta en la distribución de productos tiene que ver con la insuficiencia por parte de la flota disponible para cubrir la demanda. Este escenario hace necesaria la contratación de una flota de vehículos que cubra el restante de la demanda. Dentro del problema de ruteo de vehículos, el uso de esta flota adicional implica la creación de rutas abiertas, es decir, que no impliquen el retorno al depósito de origen. El nombre normalmente atribuido a este tipo de problema es el de “Ruteo de Vehículos con Flota Propia y Subcontratada”. Dado que la subcontratación de vehículos adicionales normalmente implica un sobrecosto, la planeación de las rutas se hace aún más relevante ya que se requiere que se utilice de manera óptima la flota propia y se minimicen los costos asociados a la flota subcontratada. Este escenario es también posible cuando el transportista tiene más de un depósito desde el cual debe de programar sus rutas, es decir, que se debe decidir qué conjunto de clientes debe ser atendido por cada depósito. La necesidad por satisfacer toda la demanda impacta directamente la planeación en la adquisición de activos, ya que los presupuestos pueden estar limitados en cuanto a la cantidad de vehículos usados como parte de la flota propia.

Por otro lado, se considera relevante mencionar un problema común se presenta en la logística de entregas de última milla, es decir, la que se enfoca en el paso final de la entrega. Los costos asociados a este último paso son por lo general los más altos, se estima que alrededor del 30% del costo de las entregas corresponden a la logística de última milla. Mientras la mayor parte del proceso de transporte de mercancías se realiza a lo largo de autopistas con baja congestión vehicular, los vehículos destinados para realizar el recorrido de última milla por lo general tienen que lidiar con el tráfico de los centros urbanos, restricciones de acceso y horarios en ciertas zonas de la ciudad. Adicionalmente, es común que las entregas puedan ser gestionadas desde diferentes centros de distribución, por lo que la planeación de las rutas se debe hacer considerando el centro de distribución más cercano convirtiéndose en un problema multidepósito. Por estas razones el transporte de última milla suele ser menos eficiente y la reducción de los costos es más difícil si se tienen en cuenta este tipo de limitaciones. Modelos de rutero que contemplan el uso de la flota subcontratada también pueden ser usados para realizar la distribución de productos en este tipo de escenario.

El principal desafío para la planeación de las rutas radica en la gran cantidad de posibles combinaciones que se pueden tener en la asignación de un vehículo a un conjunto de clientes específico. El problema se hace más complejo cuando se tienen demandas variables y una capacidad limitada de la cantidad de productos que se pueden transportar en cada vehículo. En adición, cuando se tienen problemas con restricción en el número de vehículos propios que se pueden usar, es necesario definir qué conjunto de

clientes se debe asignar a cada una de las dos flotas (propia y subcontratada), lo que implica considerar dos matrices de distancias diferentes. Esta situación hace que sea necesaria la implementación de un procedimiento intencionalmente guiado para obtener soluciones no solamente factibles, sino de calidad de manera que se minimice el costo asociado a satisfacer la demanda.

7 Justificación

El problema de ruteo de vehículos es uno de los problemas combinatoriales más tratados en la literatura, pues consiste en el diseño óptimo de rutas que usan una flota de vehículos para atender un conjunto de clientes distribuidos geográficamente. Desde que fue propuesto en (Dantzig & Ramser, 1959) se han implementado diferentes métodos de solución y se han abordado variantes agregando restricciones como la capacidad de los vehículos, problemas considerando ventanas de tiempo, la inclusión de múltiples depósitos, usando flota con capacidad heterogénea, entre otros. En el presente trabajo se propone el uso de una metodología guiada para la solución del modelo MDVRPPC, es decir, la solución del problema de ruteo de vehículos considerando rutas abiertas y cerradas, la posibilidad de cada cliente de ser atendido por más desde más de un depósito y la capacidad restringida de cada vehículo. Las aplicaciones de este tipo de modelos en el transporte y la logística son por ejemplo la entrega y recogida de productos, transporte de personas, entre otras.

El problema MDVRPPC es desarrollado para el escenario donde existe más de un depósito desde el cual se hacen los despachos, donde los clientes pueden tener demandas diferentes y donde se considera el uso de flota subcontratada (rutas abiertas), es decir, para los escenarios en que la demanda de los clientes puede superar la capacidad de los vehículos disponibles. Por lo anterior, en este trabajo se considera tanto de problema de ruteo de vehículos con múltiples depósitos (MDVRP), el problema de ruteo con restricciones de capacidad (CVRP) y el problema de ruteo con flota subcontratada (VRPPC). Dado la naturaleza combinatorial del problema de transporte, se conoce como un problema NP-Hard ya que es una extensión del problema del agente viajero o TPS por sus siglas en inglés. Esto implica que los problemas derivados o que son variantes del VRP clásico tienen una complejidad similar, es decir que, en consecuencia, el problema MDVRPPC es también un problema de complejidad NP-Hard.

Para la solución del problema MDVRPPC se propone el uso de técnicas metaheurísticas que van desde la construcción de una solución inicial, hasta una etapa de mejoramiento que permita encontrar la mejor solución posible. Las técnicas de “clusterización” son necesarias para la asignación inicial de cada cliente a un único depósito. Así mismo, el algoritmo de ahorros será usado para trazar las primeras rutas al interior de cada depósito en la construcción de una solución inicial. Para la etapa de mejoramiento se usará la “Búsqueda Local Iterada” o ILS por sus siglas en inglés.

8 Delimitación del problema

El modelo de ruteo de vehículos MDVRPPC presentado en el presente trabajo consideró las siguientes características:

1. Se tendrán restricciones de capacidad para los vehículos.

2. Flota de vehículos homogénea.

3. Se considera el problema como simétrico.

4. Las distancia entre los clientes se toma como la distancia euclidiana entre las ubicaciones.

5. Los clientes tienen demanda constante.

6. Se considera más de un depósito.

7. Los vehículos considerados como propios describen rutas cerradas.

8. Los vehículos considerados como subcontratados describen rutas abiertas, es decir, que no requieren volver al depósito de inicio.

9 Marco de Referencia

9.1 Antecedentes

El problema de ruteo de vehículos (VRP) se considera un problema clásico de optimización, este se formuló por primera vez en (Dantzig & Ramser, 1959) siendo una extensión del problema del agente viajero TSP (Travelling Salesman Problem), y desde su formulación se han propuesto múltiples variaciones mediante la adicción de nuevas condiciones al modelo más básico. Un primer cambio que integra el VRP inicial, entendiéndose como una generalización del TPS, es la utilización de varios vehículos, es decir, se considera el caso análogo del TSP con múltiples viajeros. Cambiando los viajeros por vehículos se obtiene el VRP.

CVRP (Capacited Vehicle Route Problem)

Así mismo, considerando que la capacidad de los vehículos está restringida a un valor fijo, se habla del problema CVRP (Capacited Vehicle Route Problem) y es considerado como la versión clásica del VRP. Este problema puede ser planteado definiendo un conjunto de clientes $i = \{1,2, \dots, n\}$ donde $i = 0$ corresponde al depósito. Las trayectorias que conectan a un cliente i hasta un cliente j donde $i < j$ son llamadas arcos y están asociadas a variables como la distancia, tiempo de recorrido o simplemente se puede generalizar como el costo de desplazamiento. Este costo se define como c_{ij}^k que representa el costo asociado de atravesar el arco $(i;\ j)$ en el vehículo k donde $k = \{1,2, \dots, m\}$, con m como la cantidad de vehículos disponibles. Para el caso donde el problema se considera asimétrico, se tiene que $c_{ij}^k \neq c_{ji}^k$ y simétrico cuando $c_{ij}^k = c_{ji}^k$. La definición anterior puede describirse como un grafo completo representado como $G = (V; A)$ donde $V = i$ con $i = 1; 2, \dots, n$, es decir, un conjunto que contiene a todos los clientes, y donde A corresponde al conjunto de arcos que tienen asociado un valor de c_{ij}^k. El problema CVRP consiste en la construcción en un conjunto de rutas tal que: (1) cada ruta empieza y termina en el mismo depósito, (2) la capacidad Q de cada vehículo no es excedida, (3) la demanda de cada uno de los clientes es satisfecha, (4) cada cliente debe ser visitado una sola vez, (4) la suma de todos los costos debe ser minimizada, es decir, escoger la combinación de arcos x_{ij} de tal manera que la suma de $c_{ij}^k * x_{ij}$ sea mínima. La formulación de otros problemas son consecuencia directa de la modificación del modelo CVRP mediante la adicción de restricciones y condiciones especiales, como son la utilización de vehículos con diferentes capacidades, clientes con diferentes tipos de recursos solicitados, ventanas de tiempo, variables asociadas con procesos estocásticos, diferenciación en el retorno al depósito o la inclusión de más de un depósito.

La solución del problema CVRP se ha abordado desde diferentes enfoques, entre los más reconocidos están los basados en técnicas exactas como Branch-and-Bound, Branch-and-Cut (BC), posteriormente se presentarían metodologías de solución más sofisticadas como el robust Branch-Cut-and-Price (BCP) y el Set Partitioning (SP) con cortes adicionales. En el campo de las técnicas heurísticas y metaheurísticas se puede hacer la siguiente clasificación: (1) heurísticas constructivas con fases de mejoramiento, (2) metaheurísticas con empleo de mecanismos de memoria para la búsqueda, intercambios de partes de las soluciones y perturbaciones (Subramanian, 2012).

Como se mencionó anteriormente, a partir de modificaciones del problema original del CVRP se pueden crear múltiples variaciones del problema VRP. La más relevantes y relacionadas con el modelo propuesto en el presente trabajo se muestran a continuación:

VRP Asimétrico (Asymmetric cost matrix - ACVRP)

Como se mencionó anteriormente, este problema se define cuando las distancias de (i, j) son diferentes a las distancias (j, i). Lo anterior hace que la solución debe ser encontrada a partir de una matriz asimétrica. Esta situación implica un aumento en el tiempo computacional para resolver el problema. Las restricciones de capacidad se mantienen idénticas a las del CVRP.

VRP Abierto (Open VRP - OVRP)

En esta variación los vehículos no tienen la restricción de tener que regresar al depósito, es decir, los recorridos son descritos por rutas abiertas. Esta formulación tiene aplicaciones reales como por ejemplo empresas de entrega de paquetes de transporte aéreo, donde los aviones no deben regresar al depósito o la subcontratación de la flota que solo debe responder por las entregas sin la necesidad del retorno a su origen. La modificación necesaria para transformar un problema CVRP en OVRP consiste en hacer cada uno de los costos de retornar al depósito a partir de cualquier cliente igual a cero $c_{io} = 0, \forall i \in V$ (Subramanian, 2012).

VRP con restricción de distancias (Distance-Constrained VRP - DCVRP)

En este problema se incorpora una restricción en la distancia máxima que puede recorrer cada vehículo. En este caso la restricción suele ser validada a partir del cálculo de la distancia euclidiana del recorrido de los vehículos.

VRP con flota heterogénea (Heterogeneous fleet VRP - HVRP)

Esta variación permite la inclusión de vehículos con diferentes características. Así los vehículos pueden estar diferenciados por la cantidad máxima de carga que pueden transportar, por una limitación en la distancia que pueden recorrer o por el tipo de producto que puede transportar. Estas modificaciones al modelo tienen un impacto directo en la complejidad del problema. El problema se plantea asociando a cada vehículo del conjunto $M \; con \; M \; = \{1,2, \dots m\}$, donde m es el número de vehículos disponible y donde cada uno de estos tiene una capacidad determinada Q donde $Q = \{q_1, q_2, \dots, q_m\}$. Así mismo, se establece para cada vehículo un costo diferenciado de acuerdo con sus características, por lo tanto, se tendrá que incluir una nueva variable en la función objetivo que tenga en cuenta el costo fijo de la utilización de cada vehículo en relación con la distancia recorrida. Todas las demás consideraciones y restricciones se conservan.

VRP con múltiples depósitos (Multiple Depots VRP - MDVRP)

Esta variante contempla la utilización de más de un depósito, esto crea la necesidad de determinar a partir de cuál depósito debe ser atendido cada cliente, teniendo en cuenta que cada ruta empieza y termina en el mismo depósito. Este problema se propuso inicialmente para solucionar un problema con demanda probabilística para múltiples depósitos (Tillman, 1969). Para la formulación de este problema se tiene un

conjunto $D = \{D_1, D_2, \dots, D_n\}$, correspondiente a cada depósito y un conjunto $W = \{W_1, W_2, \dots, W_n\}$, correspondiente a la capacidad de almacenamiento asociada a cada depósito D. La suma de la cantidad despacha desde un depósito D_n no deben de exceder la capacidad W_n. El problema puede plantearse inicialmente con una cantidad finita de depósitos y vehículos los cuales, como se dijo anteriormente, se asignarán desde el inicio a cada uno de los depósitos. Sin embargo, también se han formulado problemas de ruteo que incluyen la posibilidad decidir qué depósitos pueden ser o no abiertos, de manera que se minimice el costo total de satisfacer la demanda y de la apertura da cada depósito (Problema de localización CLRP). Otras variantes contemplan la restricción de la cantidad disponible de vehículos, así como la posibilidad de permitir que estos no regresen al mismo depósito del cual salieron.

VRP con recogida y entrega (Pickup-and-delivery VRP - PDVRP)

Para este problema se incluye la posibilidad de que los clientes no solo reciban productos de parte del vehículo por el cual están siendo atendidos, sino que, también pueden entregarlos. De esta manera la ocupación del vehículo está constantemente aumentando y disminuyendo durante el recorrido de su ruta. Esta carga puede, ya sea ser regresada al depósito o entregada a otro cliente. Por lo tanto, es necesario definir un nuevo conjunto de clientes $V' = \{i/i = 1,2, \dots, n\}$. La carga que entregan los clientes de V' está dada por $Q' = \{q'_1, q'_2, \dots, q'_n\}$. Este problema también es conocido y generalizado como Problema de ruteo con carga y entrega simultanea (Vehicle Routing Problem with Simultaneous Pickup and Delivery - VRPSPD).

Una variación de este problema cuando los clientes solo pueden tener productos para entregar o productos para recibir, pero no ambas situaciones al mismo tiempo. Este problema es conocido como VRP con carga y entrega mixta (Vehicle Routing Problem with Mixed Pickup and Delivery - VRPMPD). El procedimiento que se suele usar para resolver este problema es el de asignar inicialmente los clientes a los cuales se requiere realizar las entregas. En segundo lugar, se asignan mediante un proceso de inserción, el restante de los clientes a las rutas previamente construidas.

VRP con entrega dividida (Split-delivery VRP - SDVRP)

Este problema se plantea a partir de la posibilidad de que la demanda de un cliente pueda ser satisfecha mediante entregas parciales hasta completar el total de la entrega. En este problema se debe eliminar la restricción de que cada cliente sea visitado una única vez. En los casos donde la demanda de un solo cliente puede superar la capacidad de un vehículo se usa esta variante del problema de ruteo permitiendo que más de un vehículo atienda al mismo cliente.

VRP con ventanas de tiempo (VRP with TimeWindows - VRPTW)

Este tipo de problema de ruteo considera la variante de que los clientes estén disponibles para ser servidos únicamente durante un periodo específico de tiempo. Esta es una generalización del problema de ruteo con restricción de capacidad CVRP, donde la diferencia se encuentra en que existirán intervalos de tiempo denominadas generalmente como ventanas, en las cuales cualquier servicio puede empezar a partir de un intervalo de tiempo específico. El problema de ventanas de tiempo puede ser denominado “suave” si el cliente puede ser servido en un tiempo posterior al inicio del intervalo, pero asumiendo una penalización proporcional al tiempo extra que demoró en atenderse. Por otro lado, se le denomina “fuerte” cuando esta restricción no puede ser violada y los clientes deben ser atendidos en el momento

en que empieza la ventana de tiempo, en el caso en que el vehículo llegue antes del inicio de este intervalo deberá esperar hasta que el cliente esté disponible.

Una vez definido el conjunto total de clientes V, este problema consiste en representar el grafo completo con $|V| + 2$ vértices donde el depósito representa por el par de nodos 0 (empezando en el depósito) y $n + 1$ (terminando en el depósito), por lo anterior no hay arcos terminando en el vértice 0 o empezando en el vértice $n + 1$. Cada cliente i tiene una ventana de tiempo en el cual puede ser atendido $[a_i, b_i]$, así, el vehículo debe llegar antes del tiempo b_i. Si llega antes de que la ventana de tiempo abra, deberá esperar hasta el tiempo a_i. La ventana de tiempo para el depósito $[a_0, b_0]$ representa el horizonte de tiempo programado. Cada vehículo deberá volver al depósito antes de b_{n+1}. Finalmente, es necesario tener en cuenta el número de vehículos a utilizar, por lo que en caso se crea el arco $(0, n + 1)$ que indica que el vehículo no se mueve del depósito, con costos y tiempos iguales a cero.

VRP Estocástico (Stochastic VRP o SVRP)

Es definido como un problema de ruteo de vehículos donde uno o varios componentes son aleatorios. Generalmente se consideran tres tipos diferentes de SVRPs: (1) Cuando los clientes son estocásticos, es decir, que los clientes pueden estar presentes con una probabilidad P y ausentes con una probabilidad $1 - P$. (2) SVRP con demandas estocásticas lo implica que la demanda de los clientes es una variable aleatoria. (3) Cuando los tiempos de recorrido entre los clientes y el tiempo de servicio de los clientes son definidos como una variable aleatoria. Dadas las características y a la complejidad de este tipo de problemas, los desarrollos que se han hecho solo son aplicables para instancias de pocos clientes. Así mismo, es importante destacar que las técnicas para solucionar este tipo de problemas son difíciles de construir y de evaluar, sin mencionar el gasto computacional que requieren.

10 Revisión del estado del arte para el problema de ruteo con flota propia y subcontratada

El problema de ruteo de vehículos (VRP) es un problema de optimización combinatorial y de programación entera que está enfocado a encontrar el conjunto de rutas que minimicen los costos involucrados para satisfacer la demanda de un número de clientes y donde cada ruta empieza y termina en un depósito. A partir de la época de los cincuentas, donde se propuso por George Dantzig y John Ramser (Dantzig & Ramser, 1959), este problema ha sido ampliamente tratado debido a que múltiples variantes del problema pueden ser formuladas. Dichas variantes pueden ser creadas incluyendo algunas características a la formulación del modelo como los son múltiples depósitos, flota mixta, varias ventanas de tiempo, entre otras.

En el presente trabajo se estudió una variante al VRP que consiste en incluir al modelo una limitación a la cantidad de vehículos disponibles para visitar los clientes y considerando múltiples depósitos. Por lo anterior, este modelo consideró la utilización de una flota adicional para suplir la parte de la demanda que no puede ser satisfecha por la flota disponible. Así, el modelo trata el problema de múlti-depósito con flota propia y flota subcontratada o MDVRPPC por sus siglas en inglés. Lo anterior implica considerar algunas características adicionales en la formulación del problema de ruteo, lo primero es usar costos diferentes tanto las rutas en la flota propia como para la flota subcontratada. Otro ajuste necesario para implementar este escenario es el de considerar las rutas de la flota subcontratada como abiertas. De esta manera fue de interés hacer una revisión del estado del arte que describa tanto el modelo MDVRPPC como los modelos previos sobre los cuales se construye, como lo son el problema de ruteo considerando múltiples depósitos (MDVRP) y el problema de ruteo considerando múltiples depósitos sin retorno al depósito (MDOVRP).

Otro aspecto incluido en este estudio fue el de utilizar diferentes técnicas de “clusterización” que asignen conjuntos de clientes a determinados depósitos utilizando como criterio de distancia entre estos. Esto se hace para construir una solución inicial de calidad que disminuya los tiempos de convergencia. Por lo anterior, se incluyó una descripción de algunas técnicas de “clusterización” que se utilizaron en este trabajo.

Finalmente, se hizo una descripción del algoritmo de búsqueda local iterada o ILS por sus siglas en inglés, este es un procedimiento de búsqueda local que trabaja sobre soluciones ya existentes sin la necesidad de construir nuevamente una solución.

10.1 Problema de ruteo considerando múltiples depósitos (MDVRP)

El problema MDVRP es una generalización del problema de ruteo con restricción de capacidad o CVRP por sus siglas en inglés, donde más de un depósito es considerado. En este problema el vehículo empieza y termina en el mismo depósito y comúnmente la cantidad de vehículos de cada depósito es dada como

información de entrada. Un aspecto para tener en cuenta en este modelo es la metodología de asignación de cada cliente a un depósito que garantice soluciones de calidad.

Las primeras referencias al modelo MDVRP aparecen a finales de los años sesenta con la publicación de (Tillman, 1969) donde se presenta una heurística para resolver el problema con restricciones en el sistema y demanda probabilística. Otras publicaciones dejan ver la gran cantidad de contextos en los que puede ser aplicado este modelo, en el caso de (Ball, Golden, Assad, & Bodin, 1983) se describe la distribución de productos de una gran compañía química en los Estados Unidos y Canadá, el reparto de gaseosa (Bruce L Golden & Wasil, 1987). Por otra parte en (Bell et al., 1983) se realiza la gestión del inventario de gases industriales en la ubicación de los clientes combinado con la programación y despacho de vehículos. En general, existen dos tipos de técnicas para resolver este problema, las técnicas basadas en heurísticas y las técnicas exactas.

Los primeros trabajos en que se usaron técnicas exactas fueron presentados en los años ochenta por (Laporte, Nobert, & Taillefer, 1988) que logra resolver problemas de hasta 80 clientes y 8 depósitos. Otro de los trabajos que destacan basados en técnicas exactas es el publicado en (Baldacci & Mingozzi, 2009) que presenta un modelo extendido del CVRP (Problema de ruteo con restricciones de capacidad) que considera capacidad heterogénea de los vehículos denominado HVRP. Así mismo, se incluyen algunas variantes como el SDVRP (Problema de Ruteo con dependencia del sitio) y el mencionado MDVRP. El algoritmo propuesto usa tres tipos de procedimientos para establecer límites basándose en la relajación lineal y Lagrangeana en la formulación matemática, lo que reduce la cantidad de variables y permite resolver problema de hasta 200 clientes y 4 depósitos.

En (Baldacci, Bartolini, Mingozzi, & Valletta, 2011) se propone un algoritmo exacto para abordar el problema de múltiples periodos (PVRP), y consiste en asignar la combinaciones apropiadas para distribuir a los clientes y definir un conjunto de rutas de distribución para cada día en el periodo planeado. El MDVRP puede modelarse como un caso del PVRP debido a que los múltiples depósitos pueden ser considerados como diferentes periodos de tiempo.

El trabajo presentado por (Contardo & Martinelli, 2014) estudia un modelo MDVRP con restricciones de capacidad y de longitud de la ruta. Se incluyen dos formulaciones en diferentes etapas del algoritmo. La primera formulación está basada en el flujo de vehículos y es utilizada para derivar el límite inferior y realizar la fijación de variables. En segundo lugar, se tiene la formulación de partición fija que utiliza la red reducida generada en el punto anterior mediante la fijación de variables.

En cuanto a los algoritmos basados en heurísticas fueron inicialmente propuestos en los años setenta y ochenta. En (Tillman & Hering, 1971) se propone un enfoque mejorado de lo propuesto años antes en (Tillman, 1969). Así mismo, en (Tillman & Cain, 1972) se presenta un avance con respecto a (Tillman, 1969) incluyendo un procedimiento para calcular el ahorro al unir clientes a una ruta. En (Salhi & Sari, 1997) una heurística compuesta por múltiples niveles es propuesta además del diseño de dos pruebas de reducción para mejorar la eficiencia del algoritmo. De acuerdo con (Toro O., Escobar Z., & Granada E., 2015) en el primer nivel se propone una solución factible y en los siguientes dos niveles se utilizan algunas heurísticas mejorar esta solución.

En (Giosa, Tansini, & Viera, 2002) se considera el problema de múltiple depósito con ventana de tiempo (MDVRPTW). La metodología propuesta para este problema consiste en emplear estrategias de

"clusterización" para inicialmente asignar un conjunto clientes a cada depósito y posteriormente asignar las rutas.

En el artículo de (Nagy & Salhi, 2005a) se trata principalmente con el problema de ruteo considerando que los clientes pueden tanto recibir como entregar bienes al vehículo (VRPPD). El método propuesto trabaja encontrando la solución para el problema VRP y luego haciendo los ajustes para hacerla factible para el VRPPD. El algoritmo es también generalizable al caso donde se tienen múltiples depósitos. En (Crevier, Cordeau, & Laporte, 2007) proponen una extensión del problema clásico MDVRP agregando la posibilidad de que los vehículos puedan ser abastecidos durante el trayecto en otros depósitos. La metodología propone una heurística que combina un principio de memoria adaptativa con el método de búsqueda tabú y la programación entera.

En (Surekha & Sumathi, 2011) se propone una solución mediante un algoritmo genético donde los clientes correspondientes a un depósito se asignan inicialmente mediante una técnica de "clusterización". Luego una solución inicial es generada mediante el algoritmo de ahorros propuesto en (Clarke & Wright, 1964). Posterior a eso se usa un algoritmo genético, basado en un mecanismo de búsqueda paralela, más eficiente que otras técnicas como el "branch and cut", la búsqueda tabú y el recocido simulado. Por otro lado en (Jean-François Cordeau & Maischberger, 2012) una heurística paralela iterada y la búsqueda tabú son utilizadas para resolver varios problemas como VRP, SDVRP y MDVRP. La heurística combina la búsqueda tabú con un mecanismo de perturbación simple que garantiza una amplia exploración del espacio de soluciones. Así mismo, es presentada una heurística paralela que permite tomar ventaja de los múltiples núcleos del procesador.

En (Vidal, Crainic, Gendreau, Lahrichi, & Rei, 2012) se presenta un algoritmo que resuelve los problemas de MDVRP, PVRP y MDPVRP con restricciones de capacidad de los vehículos y duración de la ruta. El algoritmo usa un conjunto de operadores para la selección de los padres y cada hijo es clasificado según su factibilidad. Este algoritmo presenta un desempeño muy alto y lleva a la solución óptima de varios problemas presentes en la literatura. El trabajo presentado por (Toro, Domínguez, & Escobar, 2013) propone el uso de técnicas de "clusterización" para la creación de soluciones iniciales, además que se estudia a profundidad el impacto que tiene cada una en la función objetivo. Posterior a la generación de la solución inicial se usa el algoritmo de búsqueda local iterada para la optimización del problema.

Finalmente se presenta una revisión con los principales aportes desde finales de la década de los sesentas. La Tabla 1 se muestra que tipo de modelo fue tratado y el método empleado para su solución.

Autores	**Año**	**Tipo/aplicación**	**Método**
(Tillman, 1969)	1969	MDVRP	Algoritmo de Ahorros
(Wren & Holliday, 1972)	1972	MDVRP	Ahorros refinado
(Cassidy, P and Bennett, 1972)	1972	MDVRP (alimentos escolares)	Ahorros refinado
(Gillett & Johnson, 1976)	1976	MDVRP	Clúster y heurística de barrido
(B L Golden, Magnanti, & Nguyen, 1977)	1977	MDVRP (distribución de periódicos)	Algoritmo de ahorros modificado.
(B L Golden et al., 1977)	1983	Distribución de productos químicos	Rutear primero, agrupar después

Autores	Año	Tipo/aplicación	Método
(Laporte et al., 1988)(Perl & Daskin, 1985)	1984	MDVRP	Branch and Bound
(Perl, 1987)	1986	MDVRP (entrega de Productos de panadería)	Ahorros y Branch and Bound
(Laporte et al., 1988)	1987		Medidas de distancia y algoritmo de ahorros
(Klots, Gal, & Harpaz, 1992)	1988	MDVRP	Branch and Boundd
(Min, Current, & Schilling, 2003)	1992	MDVRP (distribución de lácteos)	Programación lineal y heuristicas
(Chao, Golden, & Wasil, 1993)	1992	MDVRP (distribución de productos de hardware)	Métodos exactos y heurísticas
(Renaud, Laporte, & Boctor, 1996)	1993	MDVRP	Grabado grabado
(Salhi & Sari, 1997)	1996	MDVRP	Búsqueda Tabú
(Jean-Fran&#xfffd;ois Cordeau, Gendreau, & Laporte, 1997)	1997	MDVRP- MDHFVRP	Heurística multinivel
(Irnich, 2000)	1997	MDVRP	Búsqueda Tabú
(Thangiah & Salhi, 2001)	2000	MDHFVRP con entregas y recogidas	Cubrimiento de conjuntos
(Tarantilis & Kiranoudis, 2002)	2001	MDVRP	Algoritmos genéticos y clusterización.
(Wassan & Osman, 2002)	2002	MDOVRP (Distribución de carne)	Umbrales basados en listas
(Giosa et al., 2002)	2002	MDVRPTW	Métodos de una y dos etapas
(Polacek, Hartl, Doerner, & Reimann, 2004)	2004	MDVRPTW	VNS
(Lim & Wang, 2004)	2005	MDVRP flota de vehículos fija	heurísticas
(Nagy & Salhi, 2005b)	2005	MDVRPMPD	Combinación de heurísticas
(Pisinger & Ropke, 2007)	2007	MDVRP	AVNS
(Dondo & Cerdá, 2007)	2007	MDHFVRPTW	Métodos exactos
(Ho, Ho, Ji, & Lau, 2008)	2008	MDVRP	Algoritmo genético
(Pepin, Desaulniers, Hertz, & Huisman, 2008)	2009	MDVSP	5 heurísticas y formulaciones
(ZHANG, TANG, & FUNG, 2011)	2011	MDVRPWRC	Formulación y SS
(Yu, Yang, & Xie, 2011)	2011	MDVRP	Colonia de hormigas
(Kuo & Wang, 2012)	2012	MDVRPLC	VNS
(Xu, Wang, & Yang, 2012)	2012	MDHVRPTW	VNS
(C.-Y. Liu, 2013)	2013	MDVRPTW	AG- ACA
(C. Liu & Yu, 2013)	2013	MDVSPTW	AG-ACO
(Subramanian, Uchoa, & Ochi, 2013)	2013	MDVRPMPD MDVRP	Matheuristica
(Venkata Narasimha, Kivelevitch, Sharma, & Kumar, 2013)	2013	Min-Max MDVRP	ACO

Autores	Año	Tipo/aplicación	Método
(Contardo & Martinelli, 2014)	2014	MDVRP	Métodos exactos
(Braekers, Caris, & Janssens, 2014)	2014	MD-H-DARP	Branch and Cut Recocido determinista
(Escobar, Linfati, Toth, & Baldoquin, 2014)	2014	MDVRP	Tabu granular
(Vidal, Crainic, Gendreau, & Prins, 2014)	2014	MDVRP- MDVFMP	
(Montoya-Torres, López Franco, Nieto Isaza, Felizzola Jiménez, & Herazo-Padilla, 2015)	2015	Estado del arte	
(Toro-ocampo, Gallego-rendón, & Franco-baquero, 2016)	2016	MDVRPPC	Modelo matemático Branch and Bound
(Afshar-Nadjafi & Afshar-Nadjafi, 2014)	2017	TDMDVRPTW	Heurísitica constructiva

Tabla 1 Revisión Cronológica MDVRP

10.2 Problemas de ruteo sin retorno al depósito (OVRP)

El problema de ruteo sin retorno al depósito es un caso particular del problema de ruteo de vehículos donde los vehículos no tienen que volver al depósito después de visitar el último cliente de la ruta. Las aplicaciones del problema sin retorno al depósito fueron mencionadas en primer lugar por (Fisher & Jaikumar, 1981; Raff, 1983) en los años ochenta cuando algunas compañías no tenían flota o esta era inadecuada para satisfacer la demanda de sus clientes. Lo anterior implica tener que usar una flota subcontratada para cubrir las rutas planeadas y que no requiere volver a punto de partida o depósito. En cuanto a costos estos están relacionados con la longitud de las rutas o el número de vehículos usados.

Los aportes al modelo OVRP permanecieron sin mayores cambios hasta la publicación de (Sariklis & Powell, 2000) donde se propone el uso de métodos de "clusterización". Luego se definen las rutas en dos etapas, primero se agrupan los clientes de acuerdo con las restricciones de capacidad y luego se utiliza una heurística denominada "Minimum Spanning Tree" con un procedimiento de penalización. Otros trabajos han sido presentados con aplicaciones en diferentes contextos y situaciones. En (Braekers, Ramaekers, & Van Nieuwenhuyse, 2016) se menciona algunas aplicaciones como la entrega de domicilios y periódicos, enrutamiento de autobuses escolares, enrutamiento de materiales en minas de carbón, envío de materiales peligrosos, entre otros.

En (Tarantilis, Ioannou, Kiranoudis, & Prastacos, 2005) se estudia un modelo OVRP utilizando un método metaheurístico de único parámetro que explota una lista de valores frontera para guiar inteligentemente

una búsqueda local avanzada. Así mismo, se realiza una comparación de este método con resultados obtenidos hasta la fecha por otras metodologías que tratan este tema.

Una versión extendida del problema de OVRP es considerar múltiples depósitos para suplir la demanda de los clientes, para este caso el modelo se define como MDOVRP. Debido a la complejidad de este tipo de problemas las técnicas más utilizadas para abordar este problema son heurísticas que se enfocan en encontrar soluciones de calidad y de manera eficiente (Dueck & Scheuer, 1990; R. Liu, Jiang, & Geng, 2014; Mirabi, Fatemi Ghomi, & Jolai, 2010; Yao, Hu, Zhang, & Tian, 2014).

En la publicación de (R. Liu et al., 2014) se presenta un trabajo que contempla rutas abiertas para el problema de múltiples depósitos MDOVRP. El algoritmo utilizado es un algoritmo genético híbrido, el cual usa una combinación de operadores de cruce global con la búsqueda local. Pocas publicaciones han sido propuestas para resolver el problema de forma exacta y el trabajo más actual para abordar el MDOVRP es el propuesto en (Lalla-Ruiz, Expósito-Izquierdo, Taheripour, & Voß, 2016) donde se expone una nueva formulación de programación lineal entera mixta adicionando algunas restricciones presentes en la literatura y adicionando algunas nuevas.

Cuando en el OVRP es necesario considerar la variante si se abre o no un depósito el problema se denomina OLRP. Los artículos más recientes que han trabajado este problema aparecen en el año 2014 con la publicación de (Wang, Du, & Ma, 2014) que propone la optimización multi objetivo para resolver el ORLP para distribución de ayudas en zonas donde se requiere la atención de desastres causados por terremotos. En este trabajo se consideraron como objetivos la minimización de los costos de distribución de la ayuda, el tiempo total de entrega y la confiabilidad de las entregas. La confiabilidad se definió como la posibilidad de los trabajadores de rescate para realizar la entrega de las ayudas. Las técnicas utilizadas en este articulo consisten en una variación del algoritmo genético denominada Non-dominated Sorting Genetic Algorithm (NSGA-II) y Non-dominated Sorting Differential Evolution (NSDE) ambas metodologías implementadas para la optimización multi objetivo. Una publicación más reciente fue hecha por (Pichka, Bajgiran, Petering, Jang, & Yue, 2018) presentando un modelo que trata con el problema de 2E-OLRP como una variante del 2E-LRP y que consiste principalmente en la inclusión de depósitos principales y depósitos satélites para satisfacer el total de la demanda. En este artículo se propuso una heurística con base matemática que inicialmente asigna los clientes a un depósito satélite de acuerdo con un criterio de distancia. En segundo lugar, se crean las rutas en la primera escala de manera aleatoria. Se considera como función objetivo minimizar el costo de apertura de los depósitos satélite y la distancia total entre depósitos satélite y clientes.

10.3 Problemas de ruteo con flota propia y subcontratada VRPPC

Es tipo de problema tiene un origen semejante al problema de OVRP en el que la flota disponible para hacer las entregas no es suficiente para satisfacer la demanda de todos los clientes. Este escenario se presenta en situaciones en las que la demanda tiene un incremento inesperado o por mantenimiento y reparaciones de la flota disponible. Por lo anterior es necesario contar con una flota adicional (subcontratada) que supla la demanda restante. De esta manera el problema VRPPC debe encontrar un conjunto de rutas cerradas y abiertas, para la flota propia y subcontratada respectivamente, que minimice los costos totales para satisfacer la demanda.

El problema de flota propia y subcontratada no ha sido ampliamente tratado en la literatura. En (Ball et al., 1983) se hacen las primeras aproximaciones, en el que a los vehículos se les asigna rutas extensas para satisfacer la demanda del conjunto de clientes. El estudio es desarrollado basándose en el problema de planeación de distribución de una compañía comercial que, debido a que tenía una gran cantidad de clientes para hacer sus entregas, requería del uso de flota propia y subcontratada. El problema se enfocó en determinar el tamaño óptimo de la ruta sin violar ninguna de las restricciones de duración de la ruta.

En (Klincewicz, Luss, & Pilcher, 1990) se enfocan en la toma decisiones, basándose el tamaño y composición de la flota, donde se determina específicamente si se utiliza flota propia o si se debe recurrir a la subcontratación de una empresa para realizar la entrega de las mercancías. Este trabajo considera la distribución geográfica de los clientes, aleatoriedad en la demanda y la utilización de un único depósito. El punto que destaca esta metodología es la solución al problema basado en la capacidad de las instalaciones y de acuerdo con su localización (Capacitated Facility Location Formulation). Así, cada cliente (sector) es atendido por un vehículo propio o subcontratado.

En el trabajo presentado en (Chu, 2005) se define el problema basándose en la necesidad que se genera al tener una demanda superior a la que puede suplir el flota de vehículos disponible, usando como recurso la subcontratación de los vehículos. En este artículo se propone un algoritmo de solución utilizando un la programación lineal binaria y utilizando una heurística que mejora el algoritmo de ahorros presentado por (Clarke & Wright, 1964).

En (Marie-Claude Bolduc, Renaud, & Boctor, 2007) se propone un procesamiento que involucra la selección de los clientes a ser servidos por la flota subcontratada, construir la solución inicial y mejorarla, construir una segunda solución inicial y mejorarla, luego la mejor solución es retenida como la solución final. En (M-C Bolduc, Renaud, Boctor, & Laporte, 2008) se propone una metaheurística diseñada para el VRPPC usando procedimientos de perturbación en las fases de construcción y mejoramiento además se realizan intercambios entre los clientes asignados a la flota subcontratada y propia.

En el trabajo presentado por (Côté & Potvin, 2009) se implementa una metodología que usa la búsqueda tabú y el encadenamiento de trayectorias. La asignación de clientes entre la flota propia y la flota subcontratada es modificada por la propiedad del encadenamiento de trayectorias.

En (R. Liu & Jiang, 2012) denominan el problema de flota propia y subcontratada como COMVRP (close-open vehicle routing problem) haciendo referencia que es necesaria la implementación de rutas abiertas y cerradas que correspondan a ambos tipos de flota. Es este trabajo se utiliza un algoritmo memético también llamado algoritmo genético hibrido que combina operadores de cruce global con búsqueda local. El comparar los resultados del algoritmo memético como los obtenidos por el CPLEX reportan un mejor desempeño logrando GAPs entre 0.3 a 13.88%.

En la publicación de (Kratica, Kostic, Tosic, Dugosija, & Filipovic, 2012) se presenta un enfoque evolucionario para resolver el problema RCSP (Routing and Carrier Selection Problem) que es equivalente a tratar con el problema VRPPC. El algoritmo genético planteado asigna a los clientes la flota propia disponible para luego ordenar de manera no decreciente los demás clientes de acuerdo a la distancia, finalmente los resultados son comparados con los casos homogéneos y heterogéneos tomados de (M-C Bolduc et al., 2008).

10.4 Problemas de ruteo multi-depósito con flota propia y subcontratada MDVRPPC

En el trabajo de (Stenger, Vigo, Enz, & Schwind, 2013) se presenta un estudio de las entregas finales, en el cual se hacen las entregas de paquetes pequeños a distancias menores a una milla. En este artículo se aborda el problema como un modelo de flota propia y subcontratada de múltiples depósitos. La metodología presentada es un algoritmo de búsqueda variable basada en el intercambio cíclico de vecindario, este algoritmo usa un mecanismo adaptativo que considera una etapa de perturbación aleatoria. Este método es utilizado con éxito para resolver el problema de MDVRPPC, así mismo es funcional para problemas relacionados con el MDVRP o el VRPPC con un único depósito. Los resultados son comparados con el algoritmo VNS estándar mostrando un mejor desempeño.

En cuanto al problema de localización con flota propia y subcontratada CLRPPC se considera como una nueva línea de investigación (Prodhon & Prins, 2014). Este problema se evidencia en las compañías en las que su nivel de actividades aumenta, esto sucede cuando aparecen actividades de mantenimiento o reparación de la flota lo que genera que no se pueda atender la demanda existente y crea la necesidad de recurrir a una flota subcontratada. Otro caso donde se presenta este problema en la logística de atención de desastres donde la demanda sobrepasa la capacidad de los organizamos dispuestos para atender estas eventualidades.

En (Toro-Ocampo, Franco-Baquero, & Gallego-Rendón, 2016) se presenta un modelo matemático exacto para la solución del problema CLRPPC, sin embargo es fácilmente adaptable al modelo MDVRPPC ya que en su formulación basta con eliminar las restricciones asociadas a la apertura de cada depósito. El modelo es probado para instancias de máximo 50 clientes dada su complejidad y el gasto computacional obteniendo un GAP del 5%, lo cual se considera aceptable si se tiene en cuenta que es un modelo exacto.

En publicaciones más recientes se tiene el trabajo realizado por (Azadeh & Farrokhi-Asl, 2017) donde se aborda el problema de rutas abiertas y rutas cerradas con flota heterogénea COMVRP para el caso de múltiples depósitos, este problema es análogo a crear un modelo que contempla la posibilidad de usar flota propia y subcontratada con restricciones de capacidad y distancia del recorrido para cada vehículo. Se asume que las demandas de cada cliente y las distancias entre nodos son determinísticas y conocidas, que cada cliente debe ser servido por un único vehículo pero que este puede servir a más de un cliente siempre y cuando no se supere su capacidad. Los vehículos de la flota propia describen rutas cerradas mientras los pertenecientes a la ruta subcontratada describirán rutas abiertas. La metodología utiliza un algoritmo genético híbrido (HGA) combinado con la técnica llamada Proceso Analítico Jerárquico (AHP) para generar las soluciones iniciales, luego el procedimiento iterativo “swap” y operadores genéticos son utilizados para mejorar la solución. Los clientes son asignados a los depósitos después de ser asignados a las rutas, luego el depósito más cercano al primer cliente de cada ruta es asignado a la misma ruta. Usando la técnica de AHP los vehículos son ordenados de acuerdo con sus características (limitaciones de capacidad, recorrido, costos de operación), así los mejor puntuados son los primeros en ser asignados a un conjunto de clientes.

10.5 Técnicas de clusterización para asignar clientes a depósitos

Las técnicas de “clusterización” son una estrategia que puede ser usada para la asignación de un conjunto de clientes a un depósito de acuerdo con un criterio, por lo general criterios de distancia. Estas técnicas tienen el propósito de mejorar la calidad de la solución inicial ya que se asume que los clientes más cercanos a un depósito son los que con mayor probabilidad serán atendidos por la flota perteneciente a ese depósito. Por lo anterior en esta sección se hace una revisión de artículos que utilizan técnicas de “clusterización” para la construcción de una solución inicial para los modelos multi-depósito.

Entre los trabajos que utilizan esta metodología destaca el presentado en (Giosa et al., 2002) en el que inicialmente se agrupa para luego aplicar el algoritmo de ahorros a cada grupo resultante en la etapa de “clusterización”. Las técnicas tratadas en este artículo usan el criterio de asignar vehículos a los depósitos de manera que la capacidad de los depósitos no sea excedida. El problema de ruteo se estudiado presentado es el MDVRPTW con el cual se mide la eficiencia de las 6 técnicas de “clusterización” implementadas. En (Nagy & Salhi, 2005a) se utiliza una variante del problema de múltiples depósitos, considerando entregas y recolección de bienes, denominado Multi-depot VRP with mixed Pickup and Delivery (MDVRPMPD). En este trabajo para encontrar la solución inicial se siguen los siguientes pasos: i) Dividir los clientes en clientes fronterizos y no fronterizos, ii) Asignar los clientes no fronterizos al depósito correspondiente, iii) Para cada subconjunto de clientes encontrar una solución factible resolviendo un problema de un único depósito, iv) Insertar los clientes de la frontera a rutas obtenidas en el paso anterior. Finalmente, mediante operadores de intercambio de clientes inter-rutas e intra-rutas se mejora la solución.

En el trabajo presentado en (Surekha & Sumathi, 2011) utilizan un metodología para asignar cada cliente a un depósito utilizando un criterio de distancia. Luego se utiliza el algoritmo de ahorros propuesto en (Clarke & Wright, 1964) para construir una solución inicial y factible, posteriormente se utiliza un algoritmo genético para mejorar dicha solución. Finalmente se tiene el trabajo de (Barreto, Ferreira, Paixão, & Santos, 2007) que implementan las técnicas de “clusterización” para resolver el problema de LRP. El aporte principal de este trabajo es el uso de distintas técnicas jerárquicas y no jerárquicas para el agrupamiento de los clientes. En definitiva, se analizaron los resultados para 19 instancias implementado cuatro versiones de una heurística y utilizando 6 medidas de distancias diferentes.

10.6 Metaheurística ILS (Iterated Local Search-Búsqueda Local Iterada)

Como se menciona en (Gallego Rendón, Toro Ocampo, & Escobar Zuluaga, 2015) el uso de técnicas metaheurísticas es necesario cuando las técnicas exactas fracasan en encontrar una solución óptima. Si bien, no garantizan el óptimo este tipo de técnicas puede llegar a soluciones de calidad en un tiempo computacional aceptable. Este tipo de metodologías pueden ir desde procedimientos simples probando indicadores la sensibilidad hasta técnicas más complejas que basados en criterios de vecindad dirigen la búsqueda en el espacio de soluciones de manera eficiente.

En esta sección se describe el tema correspondiente a la implementación de la metaheurística ILS (Iterated Local Search-Búsqueda Local Iterada). El ILS funciona perturbando soluciones óptimas locales encontradas previamente, así mediante una búsqueda local iterada se producen nuevas soluciones. El ILS tiene como característica enfocarse en un subconjunto del espacio posible de soluciones. El subconjunto está constituido por óptimos locales dados por un procedimiento de optimización. El algoritmo se

compone de cuatro pasos fundamentales, generación de la solución inicial, búsqueda local para encontrar una mejor solución, perturbación de la nueva solución para encontrar un nuevo punto inicial y finalmente parada por un criterio de aceptación. El objetivo de hacer una perturbación que tiene como objetivo escapar del actual óptimo local. La eficiencia del algoritmo está íntimamente relacionada con el procedimiento de búsqueda local escogido.

A partir de una cantidad máxima de iteraciones de se implementa un procedimiento constructivo donde en cada iteración se genera una solución. El objetivo del ILS es mejorar la solución inicial usando un procedimiento en la búsqueda local y combinándola con mecanismos de perturbación. Este proceso se puede resumir en los siguientes pasos:

- Generar una solución inicial: Esta debe realizarse por medio de una heurística constructiva.
- Búsqueda Local: Tiene el objetivo de mejorar todas las mejores soluciones obtenidas después de aplicar cualquier heurística de construcción inicial o mejoramiento.
- Perturbación: Aquí es generado un nuevo punto de inicio en el espacio de soluciones, la cual se intentará mejorar con una nueva búsqueda local.
- Criterio de aceptación: Determina cuando debe terminar la ejecución del algoritmo.

Es posible utilizar diferentes técnicas heurísticas como búsqueda local, pero al ser este un componente esencial del algoritmo del que depende su rendimiento en términos de calidad de las soluciones y esfuerzo computacional es utilizado una técnica llamada VND (Variable Neighborhood Descent), la cual utiliza un listado ordenado de vecindarios aleatorios RVND (Random Variable Neighborhood Descent). Por lo tanto, se define el conjunto $N = \{N^1, N^2, \dots, N^r\}$ que contiene los vecindarios, los cuales, si uno de los vecindarios k falla siendo mejor que la incumbente, el RVND selecciona otro vecindario aleatoriamente desde el mismo conjunto y continua con la búsqueda en el espacio de soluciones. Para este caso se consideran tres niveles diferentes en lo que se pueden realizar intercambios, N está compuesto solo por vecindarios construidos cada uno a través de estructuras de inter-ruta pertenecientes a diferentes depósitos. Para el mejoramiento se elige un vecindario del conjunto N de manera aleatoria y se determina el mejor vecino o mejor solución de aquel vecindario. En segundo lugar, se efectúa un procedimiento similar pero únicamente entre rutas pertenecientes al mismo depósito. En caso de que haya mejoramiento de la solución, es ejecutado una búsqueda local de tipo intra-ruta, pero para el caso contrario, el vecindario es removido del conjunto N.

Las estructuras inter-depósito consisten principalmente en el intercambio de clientes pertenecientes a rutas de diferentes depósitos. Para la implementación de este tipo de operadores debe considerarse que no todos los clientes son susceptibles a cambiarse, algunos de ellos están a una distancia considerable de la ruta con la cual se planea realizar el cambio, lo que puede llevar al gasto innecesario de recursos computacionales. Por esta razón los intercambios de clientes en las estructuras inter-depósito están restringidos por la distancia a las rutas a las cuales serán insertados. Las estructuras utilizadas pueden ser de tipo $Shift(k, 0)$ donde se trasladan k clientes adyacentes de una ruta asociada a un depósito a otra ruta perteneciente a un depósito diferente. De igual forma pueden realizarse intercambios del tipo $Swap(k, h)$ donde k es la cantidad de clientes adyacentes a trasladar de la primera ruta y h será el número de clientes también adyacentes pertenecientes a la otra ruta que ocuparan el lugar de los clientes extraídos de la primera ruta. Todos los intercambios de este tipo no deben sobrepasar la capacidad máxima de los vehículos ni la capacidad máxima de los depósitos. Otro operador que puede resultar útil para el caso específico del MDVRPPC es el de intercambiar vehículos de propios y subcontratados de un

depósito a otro, es decir, un operador que transforme una ruta propia en una subcontratada de un depósito y simultáneamente haga lo contrario en otro depósito, esto para conservar la cantidad de vehículos propios asignados desde un principio.

Las estructuras del tipo inter-ruta hacen referencia a los intercambios de clientes entre rutas de un mismo depósito, en este caso solo es necesario comprobar si no se sobrepasa la capacidad de los vehículos para realizar cualquier cambio. Sin embargo, para mejorar el tiempo computacional es posible usar un criterio de distancia para no transferir clientes demasiado distantes pese a que pertenecen al mismo depósito. Los operadores $Shift(k,0)$ y $Swap(k,h)$ pueden ser igualmente utilizados como operadores inter-ruta y adicionalmente puede ser implementado el operador $K-Shift(k,0)$ que consiste en el traslado de k clientes adyacentes pertenecientes a una ruta al final de otra ruta.

Finalmente, los operadores intra-ruta son usados para mejorar la solución intercambiando los clientes de una misma ruta. Los operadores pueden ser $Reinserción$, que consiste en cambiar de posición un cliente en relación los demás clientes de la ruta. $Or-Opt2$ remueve dos clientes adyacentes y los inserta en otra posición de la ruta. $Or-Opt3$ de manera similar remueve tres clientes adyacentes y los inserta en otra posición de la ruta. $2-Opt$ invierte la posición de un subconjunto de clientes mediante el intercambio de dos arcos no adyacentes.

En la metodología de búsqueda local iterada también se propone la utilización de operadores de perturbación que permitan salir de óptimos locales. Lo anterior implica desmejorar temporalmente la solución obtenida para volver a aplicar los operadores arriba descritos y explorar de manera más completa el espacio de soluciones. Para esto se suelen utilizar operadores como el $Multiple-Swap(1,1)$ y el $Multiple-Shift(1,1)$, ambos realizan intercambios de clientes entre rutas de manera aleatoria y sin tener en cuenta si la función objetivo empeora.

10.7 Modelos matemáticos en la literatura

En esta sección se describen los modelos matemáticos para abordar el problema de ruteo de múltiples depósitos con flota propia y subcontratada usados hasta la fecha. El más reciente corresponde a la publicación de (Azadeh & Farrokhi-Asl, 2017) que resuelve este problema para múltiples depósitos e involucra restricciones de distancia de viaje y capacidad de los vehículos. El modelo matemático contempla que los vehículos pertenecientes a la flota propia vuelven al depósito después de servir al último cliente, mientras que los vehículos subcontratados describen rutas abiertas. En este modelo se asume que: 1) La demandas son determinísticas y conocidas, 2) Las distancias entre los nodos son determinísticas y conocidas, 3) Cada cliente debe ser servido por un vehículo, pero cada vehículo puede servir a más de un cliente sin sobre pasar la capacidad del vehículo y 4) Los vehículos de la flota propia y subcontratada deben seguir rutas cerradas y abiertas, respectivamente. A continuación, se presenta este modelo matemático.

Conjuntos:

$D=\{d\|d=1,2,\dots,R\}$	Conjunto de Depósitos
$C=\{c\|c=1,2,\dots,N\}$	Conjunto de Clientes

$K = \{k \mid k = 1,2, \dots, P\}$	Conjunto de Vehículos
$S = \{s \mid s = 0,1\}$	Vehículo propio o subcontratado

Parámetros:

$d_i =$	Demanda del consumidor i
$Q_k =$	Capacidad del vehículo tipo k
$H_k =$	Máxima longitud permitida para el vehículo k
$I =$	Máxima cantidad de vehículos propios
$c_{ij} =$	Distancia del recorrido entre el nodo $i \in C \cup D$ y el nodo $j \in C \cup D \ y \ i \neq j$
$F_k =$	Costo fijo de uso del vehículo k
$V_k =$	Costo variable del vehículo k por unidad de distancia
$M =$	Valor muy grande

Variables de decisión:

$$x(i,j,s,k) = \begin{cases} 1 \ si \ el \ vehículo \ del \ tipo \ de \ flota \ s \ viaja \ desde \ el \ nodo \ i \in C \cup D \ y \ el \ nodo \ j \in C \cup D \\ 0 \ caso \ contrario \end{cases}$$

$$O(i,s,k) = \begin{cases} 1 \ si \ el \ vehículo \ k \ de \ del \ tipo \ de \ flota \ s \ sale \ del \ depósto \ i, \\ 0 \ caso \ contrario \end{cases}$$

$$y(i,s,k) = \begin{cases} 1 \ si \ el \ vehículo \ k \ de \ del \ tipo \ de \ flota \ s \ es \ asociado \ al \ cliente \ i, \\ 0 \ caso \ contrario \end{cases}$$

$O(i,s,k) =$	Variables continuas que representa la carga repartida del vehículo k de la flota s después de dejar al cliente i
$y(i,s,k) =$	Variable continua que representa la distancia recorrida del vehículo k de la flota s desde el depósito a un cliente i.

Formulación:

El objetivo es diseñar las rutas de servicio desde los depósitos a los clientes determinando la secuencia de clientes en cada ruta y el depósito al que pertenecen.

$$Min\ Z = \sum_{k \in K} F_k \sum_{i \in D} O(i,1,k) + \sum_{k \in K} V_k \sum_{s \in S} \sum_{i \in D \cup C} \sum_{j \in D \cup C} c_{ij} x(i,j,s,k) - \sum_{k \in K} V_k \sum_{i \in C} \sum_{j \in D} c_{ij} x(i,j,s,k) \quad (10.1)$$

Sujeto a:

$$\sum_{s \in S} \sum_{k \in K} \sum_{i \in D \cup C} x(i,j,s,k) = 1 \qquad \forall \ j \in C \quad (10.2)$$

$$\sum_{s\in S}\sum_{k\in K}\sum_{i\in D\cup C} x(j,i,s,k) = 1 \qquad \forall\, j \in C \tag{10.3}$$

$$\sum_{i\in D\cup C} x(i,j,s,k) = \sum_{i\in D\cup C} x(i,j,s,k) \quad \forall\, j \in C, s \in S, k \in K \tag{10.4}$$

$$\sum_{i\in D}\sum_{j\in D} x(i,j,s,k) = 0 \qquad \forall\, s \in S, k \in K \tag{10.5}$$

$$h(i,s,k) = 0 \qquad \forall\, i \in D, s \in S, k \in K \tag{10.6}$$

$$h(i,s,k) + c_{ij} - M\big(1 - x(i,j,s,k)\big) \le h(j,s,k) \;\; \forall i,j \in C, s \in S, k \in K \tag{10.7}$$

$$h(i,0,k) + c_{ij} - M\big(1 - x(i,j,0,k)\big) \le H_k \;\forall\, j \in D, i \in C, k \in K \tag{10.8}$$

$$h(i,1,k) + c_{ij} - M(1 - x(i,j,1,k)) \le H_k \tag{10.9}$$

$$0 \le h(i,s,k) \le \sum_{j\in D\cup C} x(j,i.s,k)H_k \;\; \forall i \in C, s \in S, k \in K \tag{10.10}$$

$$\sum_{k\in K}\sum_{i\in D}\sum_{j\in C} x(i,j,0,k) \le I \qquad \forall\, j \in C \tag{10.11}$$

$$\sum_{j\in D\cup C} x(j,i,s,k) = y(i,s,k)\;\forall i \in C, s \in S, k \in K \tag{10.12}$$

$$\sum_{i\in C} d_i y(i,s,k) \le Q_k \;\; \forall k \in K, s \in S \tag{10.13}$$

$$U(i,s,k) + d_j - M\big(1 - x(i,j,s,k)\big) \le U(j,s) \;\; \forall i,j \in C, s \in S, k \in K \tag{10.14}$$

$$d_i \le \sum_{s\in S}\sum_{k\in K} U(i,s,k) \le \sum_{s\in S}\sum_{k\in K}\sum_{j\in D\cup C} x(j,i,s,k)Q_k \;\; \forall i \in C \tag{10.15}$$

$$\sum_{k\in K}\sum_{j\in D}\sum_{i\in C} x(i,j,1,k) = 0 \tag{10.16}$$

$$\sum_{i\in D}\sum_{j\in C} x(i,j,s,k) = O(i,s,k) \;\; \forall\, k \in K, s \in S \tag{10.17}$$

$$\sum_{i\in C} x(i,j,0,k) = \sum_{i\in C} x(j,i,0,k) \;\; \forall\, j \in D, s \in S, k \in K \tag{10.18}$$

$$x(i,j,s,k) = \{0,1\} \;\; \forall i,j \in D \cup C, s \in S, k \in K \tag{10.19}$$

$$O(i,s,k) = \{0,1\} \quad \forall i \in D, s \in S, k \in K \tag{10.20}$$

$$y(i,s,k) = \{0,1\} \quad \forall i \in C, s \in S, k \in K \tag{10.21}$$

$$U(i,s,k) \geq 0 \; \forall i,j \; \in D \cup C, s \in S, k \in K \tag{10.22}$$

$$h(i,s,k) \geq 0 \; \forall i,j \; \in D \cup C, s \in S, k \in K \tag{10.23}$$

La función objetivo (10.1) minimiza el costo de atender a todos los clientes considerando su ubicación. En este modelo solamente considera el costo fijo F_k para los vehículos de la flota externa como se muestra en el primer término. En el segundo y tercer término de la función se calcula el costo variable para ambos tipos de flota. Algunas restricciones para destacar son la (10.5) que evita movimientos de vehículos entre depósitos, la (10.11) que limita el número de vehículos propios, las restricciones (10.14) y (10.15) que impiden la creación de "subtours", la (10.16) y (10.18) que garantizan que solo los vehículos de la flota propia retornen al depósito.

A continuación se presenta el modelo matemático usado en (Mirledy, Ocampo, & Echeverri, 2016) que, basado en la analogía que tiene el problema de ruteo con las redes de distribución de energía eléctrica (Lavorato, Franco, Rider, & Romero, 2012), está diseñado para resolver una gran cantidad de variantes del problema de ruteo. Estas variantes van desde los problemas más básicos como el CVRP hasta problemas de mayor complejidad como el modelo CLRPPC. Este modelo puede ser modificado y adaptado para resolver el problema MDVRPPC, por lo que fue usado para contrastar los resultados obtenidos a partir de la metodología propuesta en el presente trabajo.

Conjuntos:

I	Conjunto de centros de distribución (depósitos)
J	Conjunto de clientes
V	Conjunto de nodos $V = I \cup J$

Parámetros

O_i	Costo de apertura del centro de distribución i.
W_i	Capacidad del centro de distribución i.
F	Costo fijo asociado a cada vehículo propio utilizado en la operación.
Q	Máxima carga que puede ser transportada por un vehículo.
D_j	Demanda de cada cliente $j \in J$
c_{ij}	Costo de viajar entre los nodos i y j

P	Factor de penalización aplicado a cada arco cuando es transitado usando un vehículo subcontratado.
NV_a	Número de vehículos disponibles.

Variables

x_{ij}	Variable binaria que se activa cuando el camino entre los nodos $i, j \in V$ es recorrido por un vehículo propio.
s_{ij}	Variable binaria que se activa cuando el camino entre los nodos $i, j \in V$ es recorrido por un vehículo subcontratado.
y_i	Variable binaria que indica la apertura del centro de distribución $i \in I$.
f_{ij}	Variable binaria que define si el consumidor ubicado en el nodo $j \in J$ es atendido por una ruta que inicia en el centro de distribución $i \in I$.
z_j	Variable binaria que determina si el consumidor ubicado en el nodo $j \in J$ es el último de la ruta en atendido.
a_{ij}	Variable binaria que indica si el vehículo usa el camino desde el nodo j al centro de distribución ubicado en el nodo i.
t_{ij}	Variable continua que indica la cantidad de carga transportada entre los nodos recorridos por la flota propia i y j
l_{ij}	Variable continua que indica la cantidad de carga transportada entre los nodos recorridos por la flota subcontratada i y j

$$min = \sum_{i \in I} o_i y_i + \sum_{\substack{i \in I \\ j \in J}} F a_{ij} \sum_{i,j \in V} c_{ij} x_{ij} + \sum_{\substack{j \in J \\ i \in I}} c_{ij} a_{ij} + P \sum_{\substack{i \in V \\ j \in V}} c_{ij} s_{ij} \tag{10.24}$$

sujeto a:

$$\sum_{i \in V} x_{ij} + \sum_{i \in V} s_{ij} = 1, \qquad \forall j \in J \tag{10.25}$$

$$\sum_{k \in J} x_{jk} + \sum_{i \in I} a_{ij} = \sum_{i \in V} x_{ij}, \qquad \forall j \in J \tag{10.26}$$

$$\sum_{j \in J} x_{ij} = \sum_{j \in J} a_{ij}, \qquad \forall i \in I \tag{10.27}$$

$$\sum_{k \in J} s_{jk} \leq \sum_{i \in V} s_{ij}, \qquad \forall j \in J \tag{10.28}$$

$$x_{ij} + x_{ji} + s_{ij} + s_{ji} \leq 1, \qquad \forall i, j \in V \tag{10.29}$$

$$\sum_{\substack{i \in V \\ i \neq j}} t_{ij} + l_{ij} = \sum_{\substack{k \in V \\ k \neq j}} \left(t_{jk} + l_{jk}\right) + D_j, \qquad \forall j \in J \tag{10.30}$$

$$\sum_{\substack{i \in V \\ j \in V}} x_{ij} + s_{ij} = card(J), \qquad \forall j \in J \tag{10.31}$$

$$\sum_{i \in I} f_{ij} \leq 1, \qquad \forall j \in J \tag{10.32}$$

$$t_{ij} \leq Q x_{ij}, \qquad \forall i, j \in V \tag{10.33}$$

$$l_{ij} \leq Q S_{ij}, \qquad \forall i, j \in V \tag{10.34}$$

$$\sum_{j \in J} t_{ij} + l_{ij} \leq w_i y_i, \qquad \forall i \in I \tag{10.35}$$

$$\sum_{i \in V} s_{ij} + \sum_{k \in V} x_{jk} = 1 - z_j, \qquad \forall j \in J \tag{10.36}$$

$$1 + a_{ij} \geq f_{ij} + z_j, \qquad \forall i \in I, \forall j \in J \tag{10.37}$$

$$-\left(1 - x_{ju} - x_{uj}\right) \leq f_{ij} - f_{iu}, \qquad \forall i \in I, \forall j, u \in V \tag{10.38}$$

$$f_{ij} - f_{iu} \leq \left(1 - x_{ju} - x_{uj}\right), \qquad \forall i \in I, \forall j, u \in V \tag{10.39}$$

$$f_{ij} \geq x_{ij}, \qquad \forall i \in I, \forall j \in J \tag{10.40}$$

$$\sum_{i \in I} y_i \geq \sum_{j \in J} D_j / \sum_{i \in I} w_i, \qquad \forall i \in I \tag{10.41}$$

$$\sum_{j \in J} x_{ij} + s_{ij} \leq \sum_{j \in J} D_j / Q, \tag{10.42}$$

$$\sum_{\substack{i \in I \\ j \in J}} a_{ij} \leq N V_a, \tag{10.43}$$

$$x_{ij} \in \{0,1\}, \qquad \forall i, j \in V \tag{10.44}$$

$$s_{ij} \in \{0,1\}, \qquad \forall i, j \in V \tag{10.45}$$

$$y_i \in \{0,1\}, \qquad \forall i \in I \tag{10.46}$$

$$f_{ij} \in \{0,1\}, \qquad \forall i \in I, \forall j \in V \tag{10.47}$$

$$z_j \in \{0,1\}, \qquad \forall j \in J \tag{10.48}$$

$$a_{ij} \in \{0,1\}, \qquad \forall i \in I, \forall j \in J \tag{10.49}$$

$$t_{ij} \in R, \qquad \forall i, j \in V \tag{10.50}$$

$$l_{ij} \in R, \qquad \forall i, j \in V \tag{10.51}$$

En este modelo la función objetivo (10.24)minimiza los costos asociados a la utilización de rutas propias y rutas subcontratadas. El segundo término de la función corresponde a los arcos a_{ij} de retorno de cada una de las rutas propias y en el tercer término se aprecia el parámetro P utilizado como un factor de penalización por la utilización de rutas subcontratadas. Se destaca un conjunto de restricciones relevantes para el modelo MDVRPPC como los son la (10.27) que garantiza que la cantidad de rutas propias que salen de cada depósito sea la misma cantidad que regresa. La restricción (10.28) permite que a cada nodo si se llega con un vehículo subcontratado se salga con uno subcontratado si y solo si no es un nodo terminal. La restricción (10.30) representa el balance de flujos entre la flota propia y subcontratada. En (10.31) se garantiza la eliminación de "subtours". En (10.33) y (10.34) se limitan los flujos de los dos tipos de flotas según la capacidad de los vehículos y la variable que determina su uso. La restricción (10.36) en combinación con la (10.37) define un nodo terminal para las rutas propias. Las restricciones (10.38) y (10.39) identifican los arcos de inicio de cada ruta propia para poder asignar el arco de retorno asegurando se retorna al mismo depósito donde la ruta inició. La restricción (10.43) establece el número máximo de rutas propias según la disponibilidad de vehículos.

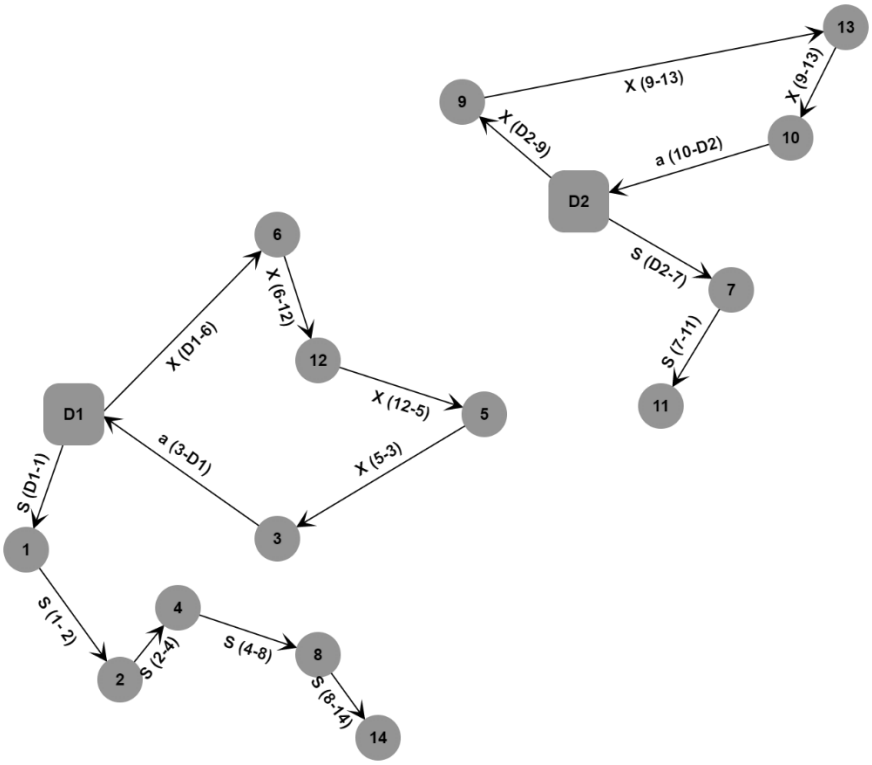


Figura 1 Estructura del MDVRPPC

De acuerdo con el modelo matemático presentado arriba, en la Figura 1 se muestra un grafo que describe el modelo MDVRPPC. Se puede considerar más de un depósito que en la figura aparecen como D_1y D_2. Después de asignar cada cliente a uno de estos depósitos se determina que clientes serán atendidos por la flota propia y cuáles serán atendidos con la flota subcontratada. Esta asignación depende de la cantidad de vehículos de los que se dispone. En la figura 1 se muestra un caso particular donde se cuentan con 2 vehículos propios y que describen rutas cerradas o que vuelven al depósito. Las variables x_{ij} se activan cuando el arco entre i y j es recorrido por un vehículo propio. La variable a_{ij} si la ruta propia termina en el nodo j y regresa al depósito i. El conjunto de variables s_{ij} se activan cuando un vehículo de la flota subcontratada es asignado para recorrer el arco entre i y j. En este caso no se activa la variable a_{ij} pues necesario que el vehículo retorne al centro de distribución.

11 Metodología

Para el desarrollo del presente proyecto se desarrollaron las siguientes etapas:

Etapa 1: Esta etapa contempló la revisión de la literatura de los problemas de ruteo de vehículos más relevantes y cuya formulación tiene una estrecha relación con la formulación del problema MDVRPPC, es decir, relacionados con el problema de ruteo con flota propia y subcontratada y considerando más de un depósito. A partir de bases de datos académicas donde se encuentran revistas indexadas nacionales e internacionales, memorias de congresos, tesis de maestrías y doctorados.

Etapa 2: Se incluyó la revisión de los modelos matemáticos que particularmente resuelven el problema MDVRPPC. Estos modelos permitieron construir una codificación del problema, definiendo el propósito de sus restricciones y función objetivo. También permitió la definición de la complejidad matemática del problema.

Etapa 3: Realización de una descripción de la metodología ILS de manera independiente para la búsqueda de soluciones del problema MDVRPCC.

Etapa 4: Construcción y desarrollo de los algoritmos requeridos para la implementación de la metodología ILS que permitieron encontrar soluciones, mostrando los tiempos computacionales y errores porcentuales con respecto a problemas solucionados con métodos exactos en instancias utilizadas en la literatura especializada.

Etapa 5: Se realizaron comparaciones de este modelo con el resultado de modelos exactos y de otros presentes en la literatura verificando la eficacia y eficiencia de la metodología propuesta.

Etapa 6: Presentación de resultados.

12 Objetivos

12.1 Objetivo General

Implementar una técnica metaheurística para la solución del problema de ruteo con flota propia y subcontratada con múltiples depósitos o MDVRPPC.

12.2 Objetivos Específicos

- Realizar una revisión del estado del arte para la solución al problema de ruteo con flota propia y subcontratada MDVRPPC utilizando técnicas metaheurísticas.
- Desarrollar un algoritmo para la ejecución de la técnica metaheurística basada en la metodología de búsqueda local iterativa ILS.
- Validar los resultados con instancias de prueba de la literatura especializada.
- Presentar resultados y metodologías.

13 Resultados Esperados

Mediante la implementación de la metodología propuesta se espera encontrar soluciones de calidad, es decir, reduciendo el GAP con respecto a los modelos exactos y en tiempos computacionalmente aceptables.

14 Adaptación del matemático para la solución del MDVRPPC

En esta sección se presenta una modificación al problema presentado en (Toro-Ocampo et al., 2016) y se describirá la función que tiene cada restricción en la solución del modelo MDVRPPC.

Para este problema solo es necesaria la simplificación de la función objetivo removiendo los dos primeros términos de la ecuación (10.24) ya corresponden a los costos de la apertura de los depósitos (en el caso del modelo CLRPPC y a los costos de apertura de las rutas respectivamente. De esta manera la función objetivo queda como se muestra en (14.1); compuesta por tres términos, donde los dos primeros corresponden a la sumatoria de los costos asociados a la flota propia y el tercero corresponde al costo total de las rutas subcontratadas. Así mismo, se considera la inclusión de la restricción (14.29) para indicar que los depósitos están abiertos o en operación y con disponibilidad de despacho.

$$min = \sum_{i,j \in V} c_{ij} x_{ij} + \sum_{\substack{j \in J \\ i \in I}} c_{ij} a_{ij} + P \sum_{\substack{i \in V \\ j \in V}} c_{ij} s_{ij} \tag{14.1}$$

sujeto a:

$$\sum_{i \in V} x_{ij} + \sum_{i \in V} s_{ij} = 1, \qquad \forall j \in J \tag{14.2}$$

$$\sum_{k \in J} x_{jk} + \sum_{i \in I} a_{ij} = \sum_{i \in V} x_{ij}, \qquad \forall j \in J \tag{14.3}$$

$$\sum_{j \in J} x_{ij} = \sum_{j \in J} a_{ij}, \qquad \forall i \in I \tag{14.4}$$

$$\sum_{k \in J} s_{jk} \leq \sum_{i \in V} s_{ij}, \qquad \forall j \in J \tag{14.5}$$

$$x_{ij} + x_{ji} + s_{ij} + s_{ji} \leq 1, \qquad \forall i, j \in V \tag{14.6}$$

$$\sum_{\substack{i \in V \\ i \neq j}} t_{ij} + l_{ij} = \sum_{\substack{k \in V \\ k \neq j}} \left(t_{jk} + l_{jk}\right) + D_j, \qquad \forall j \in J \tag{14.7}$$

$$\sum_{\substack{i \in V \\ j \in V}} x_{ij} + s_{ij} = card(J), \qquad \forall j \in J \tag{14.8}$$

$$\sum_{i \in I} f_{ij} \leq 1, \qquad \forall j \in J \tag{14.9}$$

$$t_{ij} \leq Q x_{ij}, \qquad \forall i,j \in V \tag{14.10}$$

$$l_{ij} \leq Q S_{ij}, \qquad \forall i,j \in V \tag{14.11}$$

$$\sum_{j \in J} t_{ij} + l_{ij} \leq w_i y_i, \qquad \forall i \in I \tag{14.12}$$

$$\sum_{i \in V} s_{ij} + \sum_{k \in V} x_{jk} = 1 - z_j, \qquad \forall j \in J \tag{14.13}$$

$$1 + a_{ij} \geq f_{ij} + z_j, \qquad \forall i \in I, \forall j \in J \tag{14.14}$$

$$-\left(1 - x_{ju} - x_{uj}\right) \leq f_{ij} - f_{iu}, \qquad \forall i \in I, \forall j,u \in V \tag{14.15}$$

$$f_{ij} - f_{iu} \leq \left(1 - x_{ju} - x_{uj}\right), \qquad \forall i \in I, \forall j,u \in V \tag{14.16}$$

$$f_{ij} \geq x_{ij}, \qquad \forall i \in I, \forall j \in J \tag{14.17}$$

$$\sum_{i \in I} y_i \geq \sum_{j \in J} D_j / \sum_{i \in I} w_i, \qquad \forall i \in I \tag{14.18}$$

$$\sum_{j \in J} x_{ij} + s_{ij} \leq \sum_{j \in J} D_j / Q, \tag{14.19}$$

$$\sum_{\substack{i \in I \\ j \in J}} a_{ij} \leq N V_a, \tag{14.20}$$

$$x_{ij} \in \{0,1\}, \qquad \forall i,j \in V \tag{14.21}$$

$$s_{ij} \in \{0,1\}, \qquad \forall i,j \in V \tag{14.22}$$

$$y_i \in \{0,1\}, \qquad \forall i \in I \tag{14.23}$$

$$f_{ij} \in \{0,1\}, \forall i \in I, \qquad \forall j \in V \tag{14.24}$$

$$z_j \in \{0,1\}, \qquad \forall j \in J \tag{14.25}$$

$$a_{ij} \in \{0,1\}, \qquad \forall i \in I, \forall j \in J \tag{14.26}$$

$$t_{ij} \in R, \qquad \forall i,j \in V \tag{14.27}$$

$$l_{ij} \in R, \qquad \forall i,j \in V \tag{14.28}$$

$$y_i = 1, \qquad \forall i \in I \tag{14.29}$$

En este modelo la restricción (14.2) hace que cada nodo de demanda j tenga un nodo de llegada que lo conecta a una ruta y en consecuencia a un depósito, esto se hace independientemente de si es una ruta propia o subcontratada. En (14.3) restringe la cantidad de arcos de llegada y de salida de un nodo a ser la misma, solo aplica para rutas propias. La restricción (14.4) garantiza que el número de rutas propias que salen de un depósito sea el mismo número de rutas que retornan al mismo. En (14.5) restringe a que, si se sale de un nodo con una ruta subcontratada, también se ingrese con una ruta del mismo tipo.

La restricción (14.6) evita que se recorra el mismo arco, pero en sentidos opuestos. En (14.7) garantiza el balance entre los flujos de carga entre rutas propias y rutas subcontratadas. (14.8) evita la creación de subtours. (14.9) asocia la demanda de una ruta a un depósito específico. En (14.10) y (14.11) se restringe a que el flujo máximo que transita por una ruta no supere la capacidad de los vehículos. La restricción (14.12) limita la cantidad máxima de carga que sale de los depósitos según capacidad. En (14.13) se identifica el nodo final para las rutas propias. (14.14) garantiza la existencia del arco de retorno cuando una ruta es propia. Las restricciones (14.15) y (14.16) garantiza la conexión de las rutas propias al mismo depósito tanto en su arco de salida como en su arco de retorno.

En (14.17) se garantiza que si el arco que va desde el depósito i al nodo j está activo se asocie a esa ruta con el dicho depósito mediante la activación de la variable $f_{ij} = 1$. La restricción (14.18) se deshabilita mediante la inclusión de (14.29). Con (14.19) se garantiza que existan suficientes rutas para satisfacer la demanda. (14.20) limita la cantidad de vehículos usados en las rutas propias. Las restricciones comprendidas entre la (14.21) y la (14.26) definen las $x_{ij}, s_{ij}, y_i, f_{ij}, z_j \; y \; a_{ij}$ como variables binarias. En (14.27) y (14.28) definen a las variables t_{ij}, l_{ij} como variables continuas.

15 Técnicas de clusterización

En esta sección se presentan las técnicas de "clusterización" y su aplicación al problema MDVRP. Dado que se trata únicamente de la asignación de clientes a los depósitos sin la construcción de rutas, estas técnicas pueden ser usadas en variantes del modelo como lo son el MDVRPPC y el MDOVRP. Como se mencionó en secciones anteriores las técnicas de "clusterización" se utilizan dentro de los problemas de múltiple depósito para asignar un conjunto de clientes a un depósito, esto se hace bajo el supuesto de que los clientes más cercanos a un depósito son los que tienen mayor probabilidad de ser asignados a rutas pertenecientes a dicho depósito.

Así, se hace análisis de diferentes métodos para formar clústeres a partir de una muestra de individuos. En la literatura estos métodos se denominan Métodos Jerárquicos Aglomerativos y consisten en diferentes métodos de agrupación de individuos, o en el caso del problema tratado en este trabajo, la agrupación de clientes. Estos métodos permiten tener una gran diversidad de grupos conformados por diferentes clientes según el criterio de distancia que se utilice, esto conlleva a que se pueda tener varias soluciones de agrupación de clientes, dependiendo de los grupos que se seleccionen.

Los llamados métodos jerárquicos tienen por objetivo agrupar clústeres para formar uno nuevo, de tal forma que, si sucesivamente se va efectuando este proceso de aglomeración, se minimice la distancia o incremente una medida de similitud entre los grupos formados. Por otro lado, se tiene los métodos de tipo disociativos que consiste en el procedimiento contrario al anterior, es decir, que empiezan con un conglomerado y a partir de sucesivas divisiones crea grupos más pequeños.

Los métodos aglomerativos son también llamados métodos ascendentes consideran a cada individuo como un grupo. Así, se empieza el proceso de "clusterización" uniendo en un mismo clúster los grupos menos distantes de acuerdo con criterio de distancia escogido. Este proceso se repite de manera tal que al final se conforma un único clúster que contiene a todos los individuos. En este trabajo se considera que al inicio cada cliente por separado es un clúster, y cada método es aplicado hasta que queden solo dos grupos, ya que el objetivo final es asignar los grupos de clientes a los depósitos existentes. Adicionalmente, las soluciones optimas conocidas para las instancias de problemas con múltiples depósitos nunca asignan todos los clientes a un solo depósito.

A continuación, se describen cada uno de los métodos de "clusterización" utilizados. Los métodos son consultados en (Gutiérrez, González, Torres, & Gallardo, 1994). Todos los métodos de "clusterización" se diferencian en la forma como se mide la distancia entre un par de clústeres para elegir qué grupos de clientes deben unirse. En esta sección se presentan ejemplos de la construcción de los grupos según las ecuaciones generales de distancia.

15.1 Distancia Mínima (Single Linkage)

Basándose en una matriz de distancias euclidiana entre clientes (calculada a partir de las coordenadas x y y de cada cliente), como la distancia inicial entre cada clúster, se tiene que la medida de distancia entre un par de clústeres C_i (con n_i clientes) y C_j (con n_j clientes) es:

$$d(C_i, C_j) = \min_{\substack{x_l \in C_i \\ x_m \in C_j}} \{d(x_l, x_m\} \; ; l = 1, \dots, n_i, m = 1, \dots, n_j \qquad (15.1)$$

De esta forma se une el par de clústeres que tienen el menor valor $d(C_i, C_j)$. Este proceso se repite hasta obtener solo dos grupos de clústeres. En la Tabla 2 se muestra la configuración inicial de distancias entre 7 clientes.

	A	B	C	D	E	F	G
A	0						
B	2.15	0					
C	0.7	1.53	0				
D	1.07	1.14	0.43	0			
E	0.85	1.38	0.21	0.29	0		
F	1.16	1.01	0.55	0.22	0.41	0	
G	1.56	2.83	1.86	2.04	2.02	2.05	0

Tabla 2 Matriz de distancias Inicial

Inicialmente se identifica el par de clústeres más cercanos, de la Tabla 2 se observa que los clústeres más cercanos son los conformados por los clientes (E) y (C).

Luego, como se muestra en la Tabla 3 se agrupan en un solo clúster y se recalculan las distancias al resto del grupo de clústeres. La actualización de la matriz de distancias se hace a partir de la fórmula (15.1), de esta forma la matriz queda así:

	(E,C)	A	B	D	F	G
(E,C)	0					
A	0.7	0				
B	1.38	2.15	0			
D	0.29	1.07	1.14	0		
F	0.41	1.16	1.01	0.22	0	
G	1.86	1.56	2.83	2.04	2.05	0

Tabla 3 Matriz Etapa 2 método Single Linkage

Se puede observar que solo es necesario actualizar la primera columna de la matriz de la Tabla 3, las demás distancias se conservan tal como aparecen en la matriz inmediatamente anterior (Tabla 3). Las distancias con respecto al nuevo clúster se calculan como el mínimo de la distancia de cada grupo a los grupos que conforman el clúster recién formado. A continuación, se presenta un ejemplo para el cálculo de la distancia entre el par de clústeres $C_i = (E, C)\ y\ C_j = (A)$, es de la siguiente forma:

$$d(C_i, C_j) = \min(d(E, A); d(C, A)) = \min(0.85; 0.7) = 0.7$$

De igual forma se procede para el resto de cálculo de valores en la primera columna.

En la nueva matriz se repite el procedimiento identificando el par de clústeres más cercanos, esta vez el nuevo grupo estará formado por (D) y (F). De esta forma la matriz actualizada queda así:

	(D,F)	(E,C)	A	B	G
(D,F)	0				
(E,C)	0.29	0			
A	1.07	0.7	0		
B	1.01	1.38	2.15	0	
G	2.04	1.86	1.56	2.83	0

Tabla 4 Matriz Etapa 3 método Single Linkage

De esta manera se actualiza la primera columna de la matriz de la Tabla 4, el resto de los valores pueden ser obtenidos de la matriz inmediatamente anterior. A continuación, se presenta un ejemplo para el cálculo de la distancia entre el par de clústeres $C_i = (D, F)\ y\ C_j = (E, C)$:

$$d(C_i, C_j) = \min(d(D, E); d(D, C); d(F, E); d(F, C)) = \min(0.29; 0.43; 0.41; 0.55) = 0.29$$

Seguido a esto se procede para el resto de cálculo de valores en la primera columna. Ahora los siguientes clústeres más cercanos son (E, C) y (D, F). La matriz actualizada queda así:

	(C,D,E,F)	A	B	G
(C,D,E,F)	0			
A	0.7	0		
B	1.01	2.15	0	
G	1.86	1.56	2.83	0

Tabla 5 Matriz Etapa 4 método Single Linkage

Ahora, a manera de ejemplo, se presenta el cálculo para la distancia entre el par de clústeres $C_i = (C, D, E, F)$ y $C_j = (A)$, es de la siguiente forma:

$$d(C_i, C_j) = \min(d(C, A); d(D, A); d(E, A); d(F, A)) = \min(0.7; 1.07; 0.85; 1.17) = 0.7$$

Finalmente se muestran las últimas dos matrices resultado del proceso es repetitivo:

	(A,C,D,E,F)	B	G
(A,C,D,E,F)	0		
B	1.01	0	
G	1.56	2.83	0

Tabla 6 Matriz Etapa 5 método Single Linkage

	(A,B,C,D,E,F)	G
(A,B,C,D,E,F)	0	
G	1.56	0

Tabla 7 Matriz Etapa Final método Single Linkage

La Tabla 8 muestra los grupos conformados en cada etapa del procedimiento de "clusterización". Como se puede observar inicialmente se tenían 7 grupos correspondientes a los 7 clientes. Luego se presenta el mismo conjunto de clientes después de agrupar (E, C) para un total de 6 de clústeres. Este proceso se repite hasta llegar a los resultados de la última fila donde se cuenta con clústeres.

Numero de Clústeres	Clúster	Clúster	Clúster	Clúster	Clúster	Clúster	Clúster
7	A	B	C	D	E	F	G
6	(E,C)	A	B	D	F	G	
5	(D,F)	(C,E)	A	B	G		
4	(C,E,D,F)	A	B	G			
3	(A,C,E,D,F)	B	G				
2	(A,B,C,E,D,F)	G					

Tabla 8 Todos los clústeres encontrados en método Single Linkage

En la Tabla 8 entonces resume el proceso y será la información utilizada para la asignación de clientes a los depósitos de acuerdo con su demanda acumulada y la capacidad del depósito. Este procedimiento se puede hacer por niveles, dependiendo de cuantos clústeres se necesiten para la realización de alguna tarea. La forma como se organiza la información en la Tabla 8, es equivalente a tener un dendograma. El dendograma para este ejemplo es el siguiente:

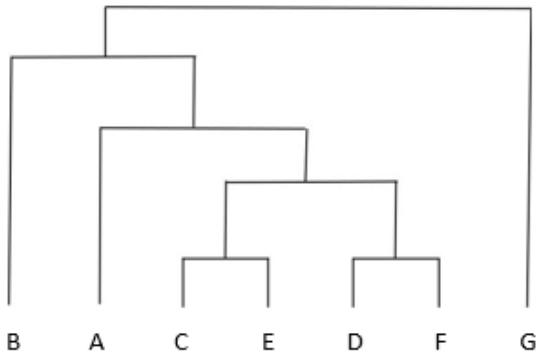


Figura 2 Dendograma método Single Linkage

En la Figura 2 se observa la representación gráfica de cómo se agruparon los clientes en el orden específico mostrado en esta metodología, y se puede observar que en la parte superior del árbol se pueden unir los dos grandes grupos de clientes en uno solo como aparecen en la última fila de la Tabla 8. No obstante, y como se explicó anteriormente, en este trabajo solo se considera la agrupación hasta mínimo dos clústeres.

15.2 Distancia Máxima (Complete Linkage)

Este método y como el general de los métodos, se inicia con la matriz de distancias euclidiana entre clientes, como la distancia inicial entre cada clúster (recuérdese que al inicio un cliente es considerado un clúster el mismo). Se tiene que la medida de distancia entre un par de clústeres C_i (con n_i clientes) y C_j (con n_j clientes) es:

$$d(C_i, C_j) = \max_{\substack{x_l \in C_i \\ x_m C_j}} \{d(x_l, x_m\} \; ; \; l = 1, \dots, n_i, m = 1, \dots, n_j \tag{15.2}$$

En este caso se unirán el par de clústeres que tengan el valor mínimo $d(Ci, Cj)$. Este proceso se repite hasta obtener solo dos grupos de clústeres. Este método es similar al anterior, pero varía en el cálculo de las distancias con los grupos que no fueron incluidos en la "clusterización" de la iteración. Se presentará la primera iteración y luego el restante cálculo de matrices para el mismo ejemplo.

En la Tabla 2 se muestra la configuración inicial de distancias entre 7 clientes, se observa que los clústeres más cercanos son los conformados por los clientes (E) y (C). Luego, se actualiza la matriz de distancias, esta vez con la ecuación (15.2):

	(E,C)	A	B	D	F	G
(E,C)	0					
A	0.85	0				
B	1.53	2.15	0			
D	0.43	1.07	1.14	0		
F	0.55	1.16	1.01	0.22	0	
G	2.02	1.56	2.83	2.04	2.05	0

Tabla 9 Matriz Etapa 2 método Complete Linkage

Como ejemplo se presenta el cálculo para la distancia entre el par de clústeres $C_i = (E, C)$ y $C_j = (A)$, es de la siguiente forma:

$$d(C_i, C_j) = \max(d(E, A); d(C, A)) = \max(0.85, 0.7) = 0.85$$

De igual forma se procede para el resto de cálculo de valores en la primera columna (solo es necesario calcular la primera columna). Los valores restantes de la matriz se obtienen de la matriz inmediatamente anterior. De nuevo se debe buscar el par de clústeres más cercanos, esta vez son (D) y (F). De esta forma la matriz actualizada queda así:

	(D,F)	(E,C)	A	B	G
(D,F)	0				
(E,C)	0.55	0			
A	1.16	0.7	0		

B	1.14	1.38	2.15	0	
G	2.05	1.86	1.56	2.83	0

Tabla 10 Matriz Etapa 3 método Complete Linkage

De nuevo solo es necesario actualizar la primera columna de la matriz de la Tabla 10, el resto de los valores pueden ser obtenidos de la matriz inmediatamente anterior calculada. Como ejemplo, el cálculo para la distancia entre el par de clústeres $C_i = (D, F)$ y $C_j = (E, C)$, es de la siguiente forma:

$$d(C_i, C_j) = \max(d(D,E); d(D,C); d(F,E), d(F,C)) = \max(0.29; 0.43; 0.41; 0.55) = 0.55$$

Puede observarse que el método es similar al *Single Linkage.* Sin embargo, se utiliza otra forma de medir la distancia entre un par de clústeres. Así mismo, en las actualizaciones de las matrices del método *Single Linkage y Complete Linkage* se obtienen valores diferentes. Si se analizamos los grupos conformados se puede notar que son los mismos clientes para ambos procedimientos, esa situación no siempre se presenta pues depende de la configuración que tengan los clientes. Así, es posible que en las primeras etapas se construyan los mismos clústeres, pero generalmente se diferencian al final del proceso de agrupación. Al finalizar el procedimiento se tiene la construcción de clústeres como se muestra en la Tabla 11.

Numero de Clústeres	Clúster	Clúster	Clúster	Clúster	Clúster	Clúster	Clúster
7	A	B	C	D	E	F	G
6	(E,C)	A	B	D	E	F	
5	(D,F)	(C,E)	A	B	G		
4	(C,E,D,F)	A	B	G			
3	(A,C,E,D,F)	B	G				
2	(A,C,E,D,F,G)	B					

Tabla 11 Todos los clústeres encontrados en método Complete Linkage

15.3 Estrategia de la distancia promedio no ponderada (Unweighted Aritmethic Average or Unweighted Average Linkage)

Este método se tiene que la medida de distancia entre un par de clústeres C_i (con n_i clientes), que a su vez está compuesto por dos clústeres C_{i1} y C_{i2} (con n_{i1} y n_{i2} elementos respectivamente), y C_j (con n_j clientes), es:

$$d(C_i, C_j) = \frac{d(c_{i1}, c_j) + d(c_{i2}, c_j)}{2} \qquad (15.3)$$

En este caso se unirán el par de clústeres que tengan el valor mínimo $d(Ci, Cj)$. Notemos que en este método no se tiene en cuenta el tamaño de ninguno de los clústeres involucrados en el cálculo, siempre se obtiene un promedio dividiendo entre dos. Este proceso se repite hasta obtener solo dos grupos de clústeres. Este método es similar al anterior, y de igual forma utiliza las matrices de distancias anteriores para calcular las nuevas distancias después de la conformación de un clúster. A continuación, se presenta un ejemplo para para dos iteraciones.

Se tiene la misma matriz de distancias de la Tabla 2, como primera etapa, se observa que los clústeres más cercanos son los conformados por los clientes (E) y (C). Luego, se actualiza la matriz de distancias, esta vez con la ecuación (15.3):

	(E,C)	A	B	D	F	G
(E,C)	0					
A	0.775	0				
B	1.455	2.15	0			
D	0.36	1.07	1.14	0		
F	0.48	1.16	1.01	0.22	0	
G	1.94	1.56	2.83	2.04	2.05	0

Tabla 12 Matriz Etapa 2 método Unweighted Average Linkage

Como ejemplo se presenta el cálculo para la distancia entre el par de clústeres $C_i = (E, C)\ y\ C_j = (A)$, es de la siguiente forma:

$$d(C_i, C_j) = \frac{d(E, A) + d(C, A)}{2} = \frac{0.85 + 0.7}{2} = 0.775$$

De igual forma se puede calcular el restante de las distancias en la primera columna (solo es necesario calcular la primera columna). Los valores para las demás columnas de la matriz se obtienen de la matriz inmediatamente anterior, ya sea ejecutando alguna operación donde estén implícitos los clústeres que se van a formar o la misma información anterior. En la segunda agrupación nuevamente se identifica el par de clústeres más cercanos, que esta vez son (D) y (F). De esta forma la matriz actualizada queda así:

	(D,F)	(E,C)	A	B	G
(D,F)	0				
(E,C)	0.42	0			
A	1.115	0.775	0		
B	1.075	1.455	2.15	0	
G	2.045	1.94	1.56	2.83	0

Tabla 13 Matriz Etapa 3 método Unweighted Average Linkage

En la Tabla 13 se actualizan los valores de la primera columna y el resto de valores pueden ser obtenidos de la matriz inmediatamente anterior, es decir, Tabla 12Tabla 13. Como ejemplo, el cálculo para la distancia entre el par de clústeres $Ci = (D, F)\ y\ Cj = (E, C)$, es de la siguiente forma:

$$d(C_i, C_j) = \frac{d(D,(E,C)) + d(F,(C,A))}{2} = \frac{0.36 + 0.48}{2} = 0.42$$

$d(F,(E,C))y\ d(F,(E,C))son\ obtenidos\ de\ la\ matriz\ de\ la$ Tabla 12

La Tabla *14* muestra los grupos conformados en cada etapa del procedimiento de "clusterización".

Número de Clústeres	Clúster	Clúster	Clúster	Clúster	Clúster	Clúster	Clúster
7	A	B	C	D	E	F	G
6	(E,C)	A	B	D	E	F	
5	(D,F)	(C,E)	A	B	G		
4	(C,E,D,F)	A	B	G			
3	(A,C,E,D,F)	B	G				
2	(A,B,C,E,D,F)	G					

Tabla 14 Todos los clústeres encontrados en método Unweighted Average Linkage

15.4 Estrategia de la distancia promedio ponderada (Weighted Arithmetic Average or Weighted Average Linkage)

A diferencia del método Unweighted Average Linkage, en este método se considera que la distancia entre dos clústeres viene definida por el promedio ponderado de las distancias de los componentes de un clúster respecto a los del otro. Sean Ci y Cj los clústeres en consideración; suponga que el clúster Ci está formado, a su vez, por otros dos clústeres, $Ci1$ y $Ci2$, con $ni1$ y $ni2$ elementos respectivamente. Entonces ni, el número de elementos de Ci , corresponde a $ni1 + ni2$ y nj al número de elementos que componen Cj. Por lo tanto, la distancia entre ambos grupos es:

$$d(C_i, C_j) = \frac{n_{i1}d(E,A) + n_{i2}d(C,A)}{n_{i1} + n_{i2}} \qquad (15.4)$$

En este caso se unirán el par de clústeres que tengan el valor mínimo $d(Ci, Cj)$. Nótese que en este método se tiene en cuenta el tamaño de los clústeres involucrados en el cálculo a diferencia del método anterior. Este proceso se repite hasta obtener solo dos grupos de clústeres. De manera análoga a los demás métodos se utilizan las matrices anteriores para actualizar la matriz de distancias en cada paso. A continuación, se presenta un ejemplo el cálculo de las dos primeras iteraciones.

Se tiene la matriz de distancias en la Tabla 2, como primera etapa, se observa que los clústeres más cercanos son los conformados por los clientes (E) y (C). Luego, se actualiza la matriz de distancias, esta vez aplicando la ecuación (15.4).

	(E,C)	A	B	D	F	G
(E,C)	0					
A	0.775	0				
B	1.455	2.15	0			
D	0.36	1.07	1.14	0		
F	0.48	1.16	1.01	0.22	0	
G	1.94	1.56	2.83	2.04	2.05	0

Tabla 15 Matriz Etapa 2 método Weighted Average Linkage

Como ejemplo se presenta el cálculo para la distancia entre el par de clústeres $C_i = (E, C)\ y\ C_j = (A)$, es de la siguiente forma:

$$d(C_i, C_j) = \frac{d(E, A) + d(C, A)}{2} = \frac{0.85 + 0.7}{2} = 0.775$$

De igual forma se procede para el resto de cálculo de valores en la primera columna (solo es necesario calcular la primera columna), todos los valores de la matriz se obtienen de la matriz inmediatamente anterior, ya sea ejecutando alguna operación donde estén implícitos los clústeres que se van a formar o la misma información anterior. De nuevo se debe buscar el par de clústeres más cercanos, esta vez son (D) y (F). De esta forma la matriz actualizada queda así:

	(D,F)	(E,C)	A	B	G
(D,F)	0				
(E,C)	0.42	0			
A	1.115	0.775	0		
B	1.075	1.455	2.15	0	
G	2.045	1.94	1.56	2.83	0

Tabla 16 Matriz Etapa 3 método Weighted Average Linkage

De nuevo solo es necesario actualizar la primera columna de la matriz de la Tabla 16, el resto de valores pueden ser obtenidos de la matriz inmediatamente anterior (Tabla 15). Como ejemplo, el cálculo para la distancia entre el par de clústeres $C_i = (D, F)\ y\ C_j = (E, C)$, es de la siguiente forma:

$$d(C_i, C_j) = \frac{d(D, E) + d(D, C) + d(F, E) + d(F, C)}{4} = \frac{0.29 + 0.43 + 0.41 + 0.55}{4} = 0.42$$

La Tabla 17 muestra los grupos conformados en cada etapa del procedimiento de "clusterización".

Número de Clústeres	Clúster	Clúster	Clúster	Clúster	Clúster	Clúster	Clúster
7	A	B	C	D	E	F	G
6	(E,C)	A	B	D	E	F	
5	(D,F)	(C,E)	A	B	G		
4	(C,E,D,F)	A	B	G			
3	(A,C,E,D,F)	B	G				
2	(A,B,C,E,D,F)	G					

Tabla 17 Todos los clústeres encontrados en método Weighted Average Linkage

15.5 Método de Centroide Ponderado

Este método puede ser utilizado con la distancia euclidiana y la distancia euclidiana al cuadrado, sin importar cual se utilice, el procedimiento es el mismo. Este método está basado en la semejanza de centros de cada clúster. Y define la distancia entre dos clústeres de la siguiente forma: A partir dos clústeres, Ci y Cj; y suponiendo que el clúster Ci está formado, a su vez, por otros dos clústeres, $Ci1$ y $Ci2$, con $ni1$ y $ni2$ elementos respectivamente. Entonces $ni1 + ni2$ es equivalente a n_i el número de elementos de Ci y nj el número de elementos que componen Cj. Entonces la distancia es:

$$d^2(C_i, C_j) = \frac{n_{i1} d^2(C_{i1} C_j)}{n_{i1} + n_{i2}} + \frac{n_{i2} d^2(C_{i2} C_j)}{n_{i1} + n_{i2}} - \frac{n_{i1} n_{i2} d^2(C_{i1} C_{i2})}{(n_{i1} + n_{i2})^2} \qquad (15.5)$$

En el caso de que se utilizara la distancia euclidiana solo es necesario cambiar el termino $d^2(Ci, Cj)$ por $d(Ci, Cj)$. Inicialmente se identifica el par de clústeres que tengan el valor mínimo $d(Ci, Cj)$ o $d^2(Ci, Cj)$ según el criterio utilizado.

Como ejemplo se presentan 5 clientes de los cuales se sabe su posición:

Sea el siguiente grupo de clientes con coordenadas (x, y):

Cliente	X	Y
A	10	5
B	20	20
C	30	10
D	30	15
E	5	10

Tabla 18 Ejemplo de clientes para métodos de centroide ponderado

En la Tabla 19 se muestra la matriz de distancias euclidiana al cuadrado:

	A	B	C	D	E
A	0				

B	325	0			
C	425	200	0		
D	500	125	25	0	
E	50	325	625	650	0

Tabla 19 Matriz Etapa 1 método Centroide Ponderado

De la Tabla 19 se observa que el par de clústeres más cercanos son el (C) y (D), que conformaran un nuevo clúster. Para este clúster se debe calcular el centroide con ayuda de las coordenadas de sus propios clientes:

$$X_{centro}^{CD} = \frac{x_c + x_D}{2} = \frac{30 + 30}{2} = 30$$

$$Y_{centro}^{CD} = \frac{y_c + y_D}{2} = \frac{10 + 15}{2} = 12.5$$

Entonces el clúster (C, D) tiene centroide $(30, 12.5)$. En caso de tener más clientes en cada clúster entonces el centroide se calcularía con todos los clientes en total n involucrados en la unión de los clústeres:

$$X_{centro} = \frac{\sum_{i=1}^{n} x_i}{n} \; y \; Y_{centro} = \frac{\sum_{i=1}^{n} y_i}{n} \tag{15.6}$$

Con este nuevo centro se actualiza la matriz en la primera columna de la matriz de distancias euclidianas al cuadrado, respecto a la distancia con los demás clústeres; los otros términos de la matriz se pueden obtener de la matriz de distancias anterior. Así en la Tabla 20 se presenta la matriz resultante:

	(C,D)	A	B	E
(C,D)	0			
A	456,25	0		
B	156,25	325	0	
E	631,25	50	325	0

Tabla 20 Matriz Etapa 2 método Centroide Ponderado

Como ejemplo, se calculará solo un elemento de la matriz anterior, el correspondiente a la distancia euclidiana al cuadrado y partiendo del a ecuación presentada en (15.7):

$$d^2(i,j) = \left(\sqrt{(i_x - j_x)^2 + (i_y - j_y)^2}\right)^2 \tag{15.7}$$

$$d^2\big((C,D),A\big) = \left(\sqrt{(X_{centro}^{CD} - X_{centro}^{A})^2 + (Y_{centro}^{CD} - Y_{centro}^{A})^2}\right)^2$$

$$d^2\big((C,D),A\big) = (30 - 10)^2 + (12.5 - 5)^2 = 456.25$$

Las restantes distancias entre clústeres con el clúster (C, D) se calcula de igual forma. Ahora, el próximo clúster estaría conformado por (A) y (E). El proceso se realiza de manera repetitiva hasta que solo queden dos clústeres. En la Tabla 21 se muestra el conjunto de clústeres resultante de aplicar este método en cada iteración.

Número de Clústeres	Clúster	Clúster	Clúster	Clúster	Clúster
5	A	B	C	D	E
4	(C,D)	A	B	E	
3	(A,E)	(C,D)	B		
2	(B,C,D)	(A,E)			

Tabla 21 Todos los clústeres encontrados en método Centroide Ponderado

Nota: La forma como se calcula la distancia entre clústeres explicada en el ejemplo anterior es equivalente a la formula presentada en (15.5). La fórmula presentada en (15.5), es una formula general, y es producto de una manipulación matemática, luego de hacer todo el proceso de este ejemplo, para presentarla en forma de numero de términos de cada clúster y distancias entre clústeres.

15.6 Método de Centroide No Ponderado o Método de la Mediana

Este método puede ser utilizado con la distancia euclidiana y la distancia euclidiana al cuadrado, sin importar cual se utilice, el procedimiento es el mismo. Este método está basado en la semejanza de centros de cada clúster, y se diferencia respecto al método del centroide ponderado en que considera que $n_{i1} = n_{i2,}$ lo que hace que el centroide del clúster C_i esté situado entre los clústeres C_{i1} y C_{i2} y con ello el centroide del clúster (C_i, C_j) esté localizado en el punto central o mediana del triángulo formado por los clústeres C_{i1} , C_{i2} y C_j. La distancia que caracteriza la distancia entre un par de clústeres es la siguiente:

$$d^2(C_i, C_j) = \frac{d^2(c_{i1}, c_j)}{2} + \frac{d^2(c_{i2}, c_j)}{2} - \frac{d^2(c_{i1}, c_{i2})}{4} \qquad (15.8)$$

A continuación, se presenta un ejemplo donde se explica el funcionamiento del procedimiento en el método de la mediana.

Se Toma como referencia la matriz de distancias de la Tabla 19. Este método se diferencia del método centroide ponderado en que siempre calcula los centros dividiendo por dos sin tener en cuenta el número de elementos en el clúster, como se observa a continuación:

Iteración 1:

De la Tabla 19, los clústeres más cercanos se unen, y estos son (C) y (D). Su centro es:

$$X_{centro}^{CD} = \frac{x_c + x_D}{2} = \frac{30 + 30}{2} = 30$$

$$Y_{centro}^{CD} = \frac{y_c + y_D}{2} = \frac{10 + 15}{2} = 12.5$$

A partir del centro del clúster se actualiza la matriz en la primera columna de la matriz de distancias euclidianas al cuadrado y respecto a la distancia de este clúster con los otros; los otros términos de la matriz se obtienen de la matriz de distancias anterior. Así la matriz queda de la siguiente forma:

	(C,D)	A	B	E
(C,D)	0			
A	456.25	0		
B	156.25	325	0	
E	631.25	50	325	0

Tabla 22 Matriz Etapa 2 método de la Mediana

A continuación, se presenta un ejemplo donde se calcula la distancia del clúster formado por (C, D) y el clúster (A) utilizando la distancia euclidiana al cuadrado (15.7):

$$d^2\big((C, D), A\big) = \left(\sqrt{(X_{centro}^{CD} - X_{centro}^{A})^2 + (Y_{centro}^{CD} - Y_{centro}^{A})^2}\right)^2$$

$$d^2\big((C, D), A\big) = (30 - 10)^2 + (12.5 - 5)^2 = 456.25$$

Las restantes distancias entre clústeres con el clúster (C, D) se calcula de igual forma.

Iteración 2:

De la Tabla 22, los clústeres más cercanos se unen, y estos son (A) y (E). Su centro es:

$$X_{centro}^{AE} = \frac{x_A + x_E}{2} = \frac{10 + 5}{2} = 7.5$$

$$Y_{centro}^{AE} = \frac{y_A + y_E}{2} = \frac{5 + 10}{2} = 7.5$$

Con este nuevo centro se actualiza la matriz en la primera columna de la matriz de distancias euclidianas al cuadrado, respecto a la distancia de este clúster con los otros; los otros términos de la matriz se pueden obtener de la matriz de distancias anterior. Así la matriz queda de la siguiente forma:

	(A,E)	(C,D)	B

(A,E)	0		
(C,D)	531.25	0	
B	312.5	156.25	0

Tabla 23 Matriz Etapa 3 método de la Mediana

A continuación, se presenta un ejemplo donde se calcula la distancia del clúster formado por (C, D) y el clúster $(A. E)$ utilizando la distancia euclidiana al cuadrado (15.7):

$$d^2\big((A,E),(\mathrm{C},\mathrm{D})\big) = \left(\sqrt{(X_{centro}^{AE} - X_{centro}^{CD})^2 + (Y_{centro}^{AE} - Y_{centro}^{CD})^2}\right)^2$$

$$d^2\big((A,E),(C,D)\big) = (30 - 7.5)^2 + (12.5 - 7.5)^2 = 531.25$$

Las restantes distancias entre clústeres con el clúster (A, E) se pueden calcular de la misma forma.

Iteración 3:

De la Tabla 23, los clústeres más cercanos se unen, y estos son (C, D) y (B). Aquí se observa la diferencia respecto al método de centroide ponderado, a pesar de que se unen tres clientes, la división se realiza por el número dos (en el método centroide ponderado explicado en el apartado anterior, se hubiera tenido que sumar cada componente de cada cliente y dividir entre número de clientes), su centro es:

$$X_{centro}^{(CD,B)} = \frac{x_{CD} + x_B}{2} = \frac{30 + 20}{2} = 25$$

$$Y_{centro}^{(CD,B)} = \frac{y_{CD} + y_B}{2} = \frac{12.5 + 20}{2} = 16.25$$

Con este nuevo centro se actualiza la matriz en la primera columna de la matriz de distancias euclidianas al cuadrado, respecto a la distancia de este clúster con los otros; los otros términos de la matriz se pueden obtener de la matriz de distancias anterior. Así la matriz queda de la siguiente forma:

	(A,E)	(B,C,D)
(A,E)	0	
(B,C,D)	382.81	0

Tabla 24 Matriz Etapa 4 método de la Mediana

Se calcula el único elemento de la matriz anterior, el correspondiente a la distancia euclidiana al cuadrado (15.7):

$$d^2\big((A,E),(\mathrm{B},\mathrm{C},\mathrm{D})\big) = \left(\sqrt{(X_{centro}^{AE} - X_{centro}^{BCD})^2 + (Y_{centro}^{AE} - Y_{centro}^{BCD})^2}\right)$$

$$d^2\big((A,E),(B,C,D)\big) = (7.5 - 25)^2 + (7.5 - 16.25)^2 = 382.81$$

Este método se explicó de forma más detallada para visualizar la diferencia entre el método de centroide ponderado y centroide no ponderado o método de la mediana.

Nota: la forma como se calcula la distancia entre clústeres explicada en el ejemplo anterior es equivalente a la formula presentada en (15.8). Esta es una formula general, y es producto de una manipulación matemática, luego de hacer todo el proceso que se presentó en este ejemplo, para presentarla en forma de distancias entre clústeres.

En la Tabla 25 se muestra el conjunto de clústeres resultante de aplicar este método en cada iteración.

Número de Clústeres	Clúster	Clúster	Clúster	Clúster	Clúster
5	A	B	C	D	E
4	(C,D)	A	B	E	
3	(A,E)	(C,D)	B		
2	(B,C,D)	(A,E)			

Tabla 25 Todos los clústeres encontrados en método de la Mediana

15.7 Método de Ward

El método de Ward, en cada etapa, une los dos clústeres para los cuales se tenga el menor incremento en el valor total de la suma de los cuadrados de las diferencias, dentro de cada clúster, de cada individuo al centroide del clúster, es decir, minimiza la siguiente expresión en cada clúster k:

$$E_k = \sum_{i=1}^{nk} \sum_{j=1}^{n} \left(x_{ij}^k - m_j^k\right)^2 \qquad (15.9)$$

Donde:

1. E_k es la suma de cuadrados de los errores del clúster k (distancia euclidiana al cuadrado entre cada individuo del clúster *k* a su centroide).
2. $\left(x_{ij}^k - m_j^k\right)^2$ representa la distancia euclidiana al cuadrado de un cliente del clúster k. Donde x_{ij}^k y m_j^k son vectores de coordenadas de cliente en el clúster k y centroide en el clúster k.

El algoritmo en cada paso calcula todos los posibles errores, y une el par de clústeres con el que se obtenga el menor error. En el siguiente ejemplo se observa el comportamiento del algoritmo.

Iteración 1.

Basado en las coordenadas de la Tabla 18 (5 clientes). Este método inicia con todos los clientes como un único clúster, ahora se debe analizar que par de clústeres se deben agrupar de los n clientes. Se debe hacer el cálculo del error E_k para cada par de clientes posibles. En esta primera iteración el número de casos a analizar se obtiene con la siguiente pregunta ¿Cuántas combinaciones de 2 clústeres se obtienen en el conjunto de n clústeres posibles? (al principio son n clientes o n clústeres, y siempre se forman grupos de dos clústeres por iteración).

$$\#Casos\ a\ analizar = \binom{5}{2} = \frac{5!}{2! * (5-2)!} = 10$$

El cálculo de los errores en la primera iteración para cada posible clúster que se pueda formar se muestra en la Tabla 26.

$Partición$	$Centroides$	Ek	E	ΔE
$(A,B),C,D,E$	$C_{AB} = (15,12{,}5)$	$E_{AB} = 162{,}5$ $E_C = E_D = E_E = 0$	162,5	162,5
$(A,C),B,D,E$	$C_{AC} = (20,7{,}5)$	$E_{AC} = 212{,}5$ $E_B = E_D = E_E = 0$	212,5	212,5
$(A,D),B,C,E$	$C_{AD} = (20,10)$	$E_{AD} = 250$ $E_B = E_C = E_E = 0$	250	250
$(A,E),B,C,D$	$C_{AE} = (7{,}5,7{,}5)$	$E_{AE} = 25$ $E_B = E_C = E_D = 0$	25	25
$(B,C),A,D,E$	$C_{BC} = (25,15)$	$E_{BC} = 100$ $E_A = E_D = E_E = 0$	100	100
$(B,D),A,C,E$	$C_{BD} = (25,17{,}5)$	$E_{BD} = 62{,}5$ $E_A = E_C = E_E = 0$	62,5	62,5
$(B,E),A,C,D$	$C_{BE} = (12{,}5,15)$	$E_{BE} = 162{,}5$ $E_A = E_C = E_D = 0$	162,5	162,5
$(C,D),A,B,E$	$C_{CD} = (30,12{,}5)$	$E_{CD} = 12{,}5$ $E_A = E_B = E_E = 0$	12,5	12,5
$(C,E),A,B,D$	$C_{CE} = (17{,}5;\ 10)$	$E_{CE} = 312{,}5$ $E_A = E_B = E_D = 0$	312,5	312,5
$(D,E),A,B,C$	$C_{DE} = (17{,}5;\ 12{,}5)$	$E_{DE} = 325$ $E_A = E_B = E_C = 0$	325	325

Tabla 26 Etapa 1 método Ward

A continuación de se presenta un ejemplo donde se calculan los errores para el primer caso de la Tabla 26, se agrupan (A,B) como un clúster. Entonces, primero se calcula el centroide de (A,B):

$$X_{AB} = \frac{10+20}{2} = 15$$

$$Y_{AB} = \frac{20+5}{2} = 12.5$$

Ahora el error para el clúster (A, B):

$$E_{AB} = (X_A - X_{AB})^2 + (Y_A - Y_{AB})^2 + (X_B - X_{AB})^2 + (Y_B - Y_{AB})^2$$

$$\mathrm{E}_{AB} = (10 - 15)^2 + (5 - 12{,}5)^2 + (20 - 15)^2 + (20 - 12{,}5)^2$$

$$\mathrm{E}_{AB} = 162.5$$

En este primer caso los errores E_C, E_D y E_E son cero, porque la distancia de cada uno de los clientes en cada clúster a su propio centro es cero, ya que están conformados por un único cliente, es decir, para E_C.

$$E_c = (X_c - X_c^{centro})^2 + (Y_c - Y_c^{centro})^2$$

$y\ como: X_c = X_c^{centro}\ y\ \ Y_c = Y_c^{centro}$

Para cada caso se hace el mismo procedimiento, y en esta primera iteración son 10 casos.

Luego, de tener todos los casos por completo, se suman los errores en cada caso y se totalizan, se analiza quien obtuvo el menor resultado de la suma de errores, y estos clústeres son los que deberían de unirse. En este ejemplo se deben unir los clústeres (C) y (D), conformando el clúster (C, D).

Iteración 2.

En esta iteración se tienen 4 clústeres, ya que anteriormente se agruparon dos, entonces se deben calcular cuántos casos se deben analizar, de esta forma se tiene:

$$\#Casos\ a\ analizar = \binom{4}{2} = \frac{4!}{2! * (4-2)!} = 6$$

El cálculo de los errores en la segunda iteración para cada posible clúster que se puede formar como se muestra a continuación:

$Partición$	$Centroides$	Ek	E	ΔE
$(A, C, D), B, E$	$C_{ACD} = (23{,}33, 10)$	$E_{ACD} = 316{,}66$ $E_B = E_E = 0$	316,66	304,16
$(B, C, D), A, E$	$C_{BCD} = (26{,}66, 15)$	$E_{BCD} = 116{,}66$ $E_A = E_E = 0$	116,66	104,16
$(C, D, E), A, B$	$C_{CDE} = (21{,}66, 11{,}66)$	$E_{CDE} = 433{,}33$ $E_A = E_B = 0$	433,33	420,83
$(A, B), (C, D), E$	$C_{AB} = (15, 12{,}5)$ $C_{CD} = (30, 12{,}5)$	$E_{AB} = 162{,}5$ $E_{CD} = 12{,}5$ $E_E = 0$	175	162,5
$(A, E), (C, D), B$	$C_{AE} = (7{,}5, 7{,}5)$ $C_{CD} = (30, 12{,}5)$	$E_{AE} = 25$ $E_{CD} = 12{,}5$ $E_B = 0$	37,5	25

$(B,E),(C,D),A$	$C_{BE} = (12{,}5, 15)$ $C_{CD} = (30, 12{,}5)$	$E_{BE} = 162{,}5$ $E_{CD} = 12{,}5$ $E_A = 0$	175	162,5

Tabla 27 Etapa 2 método Ward

Como ejemplo se calcularán los errores para el primer caso de la Tabla 27, donde se agrupan (A, C, B) como un clúster. Entonces, primero se calcula el centroide de (A, C, D):

$$X_{ACD} = \frac{10 + 30 + 30}{3} = 23.33$$

$$Y_{ACD} = \frac{5 + 10 + 15}{3} = 10$$

Ahora el error para el clúster (A, C, D):

$$E_{ACD} = (X_A - X_{ACD})^2 + (Y_A - Y_{ACD})^2 + (X_C - X_{ACD})^2 + (Y_C - Y_{ACD})^2 + (X_D - X_{ACD})^2 + (Y_D - Y_{ACD})^2$$

$$E_{ACD} = (10 - 23.33)^2 + (5 - 10)^2 + (30 - 23.33)^2 + (10 - 10)^2 + (30 - 23.33)^2 + (15 - 10)^2$$

$$E_{ACD} = 316.66$$

De otro lado los errores E_B, y E_E son cero por solo clústeres conformados por un único cliente.

Para cada caso se hace el mismo procedimiento, y en esta primera iteración son 6 casos. En esta iteración deberían unirse según la Tabla 27, los clústeres (C) y (D).

En la Tabla 28 se muestra el conjunto de clústeres resultante de aplicar este método en cada iteración.

Numero de Clústeres	Clúster	Clúster	Clúster	Clúster	Clúster
5	A	B	C	D	E
4	(A,E)	B	C	D	
3	(C,D)	(A,E)	B		
2	(B,C,D)	(A,E)			

Tabla 28 Todos los clústeres encontrados en método Ward

16 Asignación de clientes con los métodos de clusterización y aplicación de la técnica ahorros a las configuraciones de clústeres

Después de implementar las técnicas de "clusterización" es necesario definir una metodología para la asignación de los grupos a cada depósito. Por lo anterior, en esta sección se describe el procedimiento presentado en (Ocampo, Castaño, & Zuluaga, 2016) donde se aplican las técnicas de clústeres explicadas en la anterior sección. Por otro lado, se explica la técnica heurística de ahorros (*Savings Algorithm) presentada en* (Clarke & Wright, 1964), esta técnica encuentra una ruta de calidad para un conjunto de clientes con un único depósito, existen más técnicas de ruteo, sin embargo se explica esta que fue la que se implementó en el presente trabajo. Es importante aclarar que en el presente trabajo se utilizó una modificación al algoritmo de ahorros que considera la restricción de disponibilidad de vehículos propios. Esto permite generar rutas abiertas que representarán a la flota subcontratada, pero que funciona de igual forma que el planteamiento original de la técnica de ahorros.

Como se explicó arriba, esta sección tiene el objetivo de presentar el algoritmo propuesto para la asignación de clientes a cada un depósito, partiendo de los métodos considerados en la anterior sección. También presentar un algoritmo de búsqueda exhaustiva de asignación de clústeres en depósitos con base en la técnica de ahorros.

16.1 Asignación de clientes a cada depósito

Suponga que la Tabla 29 muestra los grupos conformados después de haber aplicado cualquiera de las técnicas presentadas en la sección de **Técnicas de clusterización**:

Nº Clúster	Fila	Clústeres-Columna 1	Clústeres-Columna 2	Clústeres-Columna 3	Clústeres-Columna 4	Clústeres-Columna 5	Clústeres-Columna 6	Clústeres-Columna 7
7	1	A	B	C	D	E	F	G
6	2	(E,C)	A	B	D	E	F	
5	3	(D,F)	(C,E)	A	B	G		
4	4	(C,E,D,F)	A	B	G			
3	5	(A,C,E,D,F)	B	G				
2	6	(A,B,C,E,D,F)	G					

Tabla 29 Todos los clústeres encontrados

Algunas consideraciones importantes sobre la Tabla 29 deben ser hechas antes de explicar el algoritmo de asignación de clientes a los depósitos:

1. A cada clúster de cada columna y cada fila de la matriz es posible calcular su centroide.

2. Para cada clúster de cada columna y cada fila de la matriz es posible generar una lista con la información de los depósitos más cercanos. La cercanía se define como la distancia de cada depósito al centroide de cada clúster.
3. En general se puede visualizar, que los grupos de clústeres que tienen más clientes son los que estén en las últimas filas de la matriz (en el ejemplo, la última fila tiene 2 clústeres, uno contiene casi todos los clientes y el segundo únicamente tiene 1), y a medida que se sube hasta la primera fila de la matriz, los grupos de clústeres aumentan, pero también se disminuyen en número de clientes (en la primera fila se tienen 7 clústeres, pero cada uno tiene solo un cliente).
4. Obsérvese que la matriz en cada fila se tiene diferente dimensión.
5. Para cada clúster de la matriz es posible determinar su demanda sumando las demandas individuales de los clientes que lo conforman.

A partir de las consideraciones anteriores se describe el algoritmo de asignación de clientes a cada depósito como sigue:

1. Seleccionar un punto de búsqueda inicial; ejemplo: el clúster de la última fila de la matriz, primera columna.
2. Hacer una variable: N°_Clientes_Asignados = 0.
3. Analizar si el clúster seleccionado puede ser asignado al depósito más cercano en su lista depósitos-más-cercanos (verificando siempre que el depósito tenga espacio para la demanda total del clúster, y que no se repitan clientes en el depósito, si el clúster que va a ser asignado a un depósito tiene clientes que ya están incluidos en el depósito, el clúster se rechaza por completo y no puede ser asignado). Si no puede ser asignado, buscar en la lista depósitos-más-cercanos el siguiente en la lista, hasta que asigne el clúster con el grupo de clientes a algún depósito. Se tienen los siguientes casos: Si logra asignar el clúster a algún depósito ir al paso 4, si no logra asignar el clúster a algún depósito ir al paso 5.
4. Actualizar: N°_Clientes_Asignados = N°_Clientes_Asignados + N°_Clientes_en_Clúster; y actualizar la capacidad del depósito (la cual disminuye en la demanda total del clúster). Ir al paso 5.
5. Preguntar: ¿N°_Clientes_Asignados = N°_de_Clientes_Total?, Si la respuesta es afirmativa, entonces parar, ya se han asignado todos los clientes. Si la respuesta es negativa pasar al paso 6.
6. Actualizar columna de asignación de clústeres, seleccionando el clúster de la siguiente columna de la fila actual. Si no existe clúster en esa columna y fila ir al paso 7, de lo contrario ir al paso 3.
7. Actualizar fila y columna de búsqueda. La fila de búsqueda es la que sigue en orden descendente, y la columna de nuevo es la columna 1. Seleccionar el clúster de esa fila y columna actual e ir al paso 3.

16.2 Algoritmo de búsqueda exhaustiva con la técnica ahorros

Luego de presentar el algoritmo para asignar clústeres con los clientes a los depósitos, se puede elaborar un algoritmo de búsqueda exhaustiva para buscar la mejor asignación de clústeres en los depósitos

utilizando el algoritmo de ahorros. Es importante mencionar que esta solución no es necesariamente la solución óptima, sino que dentro de todas las posibles soluciones iniciales se puede tomar la que implique menor costo usando el algoritmo de ahorros.

Supóngase que se tiene una solución de clústeres según el procedimiento descrito en la sección 15, entonces sea:

P_Fila: primera fila de la matriz solución

U_Fila: última fila de la matriz solución

P_Columna: primera columna de la fila actual

El algoritmo es el siguiente:

1. Hacer punto de inicio de búsqueda: $[U\_Fila, P\_Columna]$.
2. Con el punto de búsqueda aplicar algoritmo de asignación de clientes.
3. Paso el caso del MDVRPPC, utilizar la metodología de asignación de vehículos a clientes descrita en la siguiente sección.
4. Resolver con la técnica ahorros (modificada para flota propia y subcontratada) la asignación obtenida el problema de ruteo.
5. Guardar solución de la técnica ahorros y actualizar solución incumbente.
6. Actualizar punto de inicio de búsqueda, hacer: $U\_Fila = U\_Fila - 1$, y de nuevo pararse en $P\_Columna$.
7. Repetir pasos 2, 3, 4, y 5 hasta que $U\_Fila = P\_Fila$.

El algoritmo anterior hace una búsqueda exhaustiva para encontrar el mejor punto de arranque en número de clústeres.

16.3 Metodología de asignación de vehículos a cada depósito

En esta sección se presenta un modelo de programación no lineal para asignación de vehículos a los depósitos. Teniendo en cuenta las restricciones de flota propia que tienen los modelos como el MDVRPPC, es decir, que la cantidad de vehículos disponibles para realizar recorridos cerrados es limitada. Así, es necesaria la construcción de una metodología de optimice la distribución de los vehículos en cada depósito. Por esta razón se propone minimizar las diferencias entre la demanda de cada uno de los depósitos y la capacidad total de los vehículos asignados a estos. Así, el siguiente modelo fue desarrollado para asignar los vehículos a cada depósito.

$$Z_{min} = (CvX_1 - D_{dep1})^2 + (CvX_2 - D_{dep2})^2 + \cdots + (CvX_m - D_{depm})^2 \qquad (16.1)$$

$S.a.$

$$X_1 + X_2 + \dots X_m = Nv \quad (16.2)$$

$$X_1, X_2, \dots X_m = Entero \quad (16.3)$$

Donde:

La variable de decisión $X_i \quad con\ i = 1, 2, \dots m$ corresponde la cantidad de vehículos propios asignados al depósito i.

Nv corresponde a número total de vehículos disponible.

Cv corresponde a la capacidad de los vehículos.

D_{depi} corresponde a la demanda total asignada al depósito i

Como se puede observar la función objetivo (16.3) minimiza las diferencias entre la demanda de los clientes asignados a cada depósito y la capacidad total de los vehículos asignados. La primera restricción (16.2) garantiza que la totalidad de los vehículos sea asignada y la segunda restricción (16.1) garantiza que cantidades enteras de vehículos sean asignadas a los depósitos. Este modelo debe ser ejecutado únicamente una vez por cada instancia analizada y dado que, por lo general, el número de depósitos no supera los 10, el tiempo computacional no es significativo con respecto al tiempo de ejecución total.

17 Algoritmo de Ahorros modificado para la generación de la solución inicial

Desde su formulación en la década de los sesentas por (Clarke & Wright, 1964) el algoritmo de ahorros se ha constituido en una de la metodologías más usadas en las técnicas metaheurísticas para la construcción de soluciones iniciales en los problemas de ruteo de vehículos. En este caso representa la metodología base para la aplicación de la búsqueda local iterada o ILS, ya que parte del principio de usar soluciones de calidad encontradas previamente para mejorarlas mediante estrategias de perturbación. En este trabajo se utilizará esta metodología implementando las modificaciones al algoritmo de ahorros clásico propuestas por (Chu, 2005) y que permiten la construcción tanto de rutas cerradas como abiertas, características propias de los modelos con flota propia y subcontratada.

En su formulación original el algoritmo de ahorros está diseñado para construir soluciones factibles cuando el número de vehículos no está definido, sino que es una variable de decisión. En general esta técnica se encarga de construir la menor cantidad de rutas posibles usando como criterio el ahorro que significa la inclusión de un nodo a una ruta específica. Por otro lado, la modificación que se hace en (Chu, 2005) implica la asignación de clientes en varias etapas, en primer lugar se asignan los clientes que serán atendidos con flota propia. En segundo lugar, se ejecuta un procedimiento de inserción para tratar de incluir los demás clientes a las rutas propias ya construidas y finalmente se crean rutas abiertas con los clientes que faltan por ser asignados. A continuación, se describen los pasos principales para la implementación de esta metodología.

17.1 Procedimiento de selección

Inicialmente se requiere definir el conjunto de clientes que serán atendidos por la flota subcontratada. Lo primero es verificar que la capacidad de la flota propia es suficiente para satisfacer el total de la demanda. Si la respuesta es sí, se puede saltar este paso y continuar directamente con el siguiente.

El procedimiento parte del supuesto de que los costos de atender a un conjunto de clientes siempre son mayores si son atendidos con flota subcontratada. A partir de este supuesto se deben ordenar de manera ascendente los clientes según el costo de atenderse con la flota subcontratada y escoger los de menor costo. A continuación, se describe de manera detallada dicho procedimiento.

1. Calcular la demanda total de todos los clientes.
2. Calcular la capacidad total de la flota propia.
3. Si la demanda de todos los clientes es más grande que la capacidad de los vehículos continuar con al paso 4. En caso contrario omitir este procedimiento.
4. Calcular el total de la demanda insatisfecha restando de la demanda total, la capacidad de la flota propia.
5. Ordenar los clientes de manera ascendente según el costo de ser atendidos por la flota subcontratada.

6. Sumar la demanda de cada cliente hasta que su valor sea mayor de que la demanda insatisfecha. Estos clientes deben ser atendidos por la flota subcontratada; los clientes restantes serán asignados a los vehículos propios.

17.2 Construcción de rutas propias

En este paso se ejecuta el algoritmo de ahorros clásico, pero realizando una modificación en el cálculo de la matriz de ahorros. Así, los ahorros ahora son calculados asumiendo que en principio todos los clientes (de manera individual) son atendidos por vehículos subcontratados. El ahorro entonces estará determinado por la diferencia entre atender dos clientes con dos vehículos subcontratados y atenderlos con un solo vehículo propio. En la Figura 3 se muestra el cálculo del ahorro para esta primera etapa.

Donde:

$S_{ij} =$ Ahorros de consolidar los clientes i y j en la misma ruta.

$LTL_i =$ Costo total de servir con una ruta subcontratada al cliente i

$TL_{ij} =$ El costo total de un vehículo propio que visita al cliente i, luego visita al cliente j y finalmente vuelve al depósito.

$d_{ij} =$ *Distancia entre el cliente i y el cliente j.*

$F =$ *Factor de sobre costo por la utilización de la flota subcontratada.*

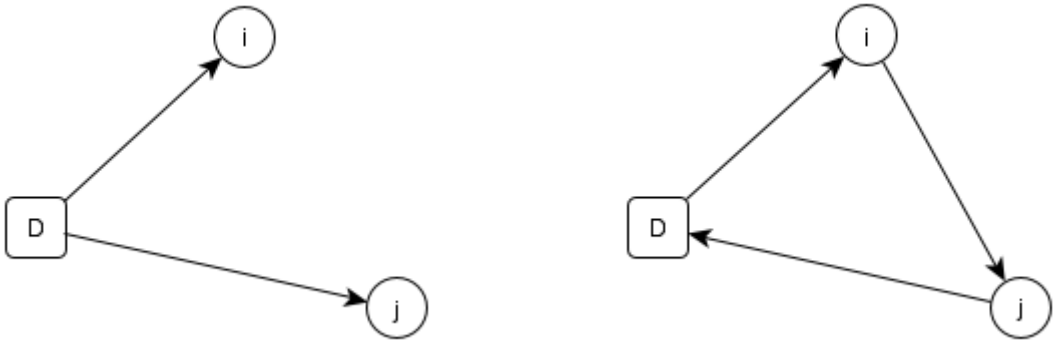


Figura 3 Calculo del Ahorro Caso 1

Costo Independiente:
$LTL_i + LTL_i$

Costo Consolidado:
TL_{ij}

Total, Ahorro:

$$S_{ij} = LTL_i + LTL_i - TL_{ij} = F * d_{Di} + F * d_{Dj} - (d_{Di} + d_{ij} + d_{jD})$$

$$S_{ij} = d_{Di} * (F - 1) + d_{jD} * (F - 1) - d_{ij}$$

$$S_{ij} = (d_{Di} + d_{jD}) * (F - 1) - d_{ij} \qquad (17.1)$$

El procedimiento para la construcción de las rutas en este paso se presenta más detallada a continuación:

1. Calcular los ahorros para cada par de clientes de acuerdo con lo mostrado en la Figura 3.
2. Ordenar los ahorros de manera descendente y empezando en lo más alto de la lista hacer lo siguiente.
3. Encontrar un enlace factible de la lista que pueda ser usado para extender uno de los dos extremos de la ruta que actualmente se está construyendo.
4. Si la ruta no puede seguir siendo extendida, finalizar la ruta. Seleccionar el primer enlace factible de la lista para empezar una nueva ruta.
5. Repetir los pasos (3) y (4) hasta que no sea posible seleccionar más enlaces.
6. Extraer todos demás clientes como rutas subcontratadas de un solo cliente y la rutas de más de un cliente que se construyeron hasta el paso (5).

17.3 Inserción de clientes a rutas previamente construidas

En esta etapa se toman los clientes que no fueron asignados en la etapa anterior y se tratan de insertar a las rutas propias ya construidas. Esto implica dejar de usar un vehículo de flota subcontratada y asignarlo a una ruta propia, movimiento que debería tener un ahorro ya que se parte del supuesto de que utilizar vehículos propios es menos costoso. No obstante, no siempre se reduce el costo al hacer este tipo de inserciones, por lo que solo se consideran los movimientos donde el ahorro es positivo.

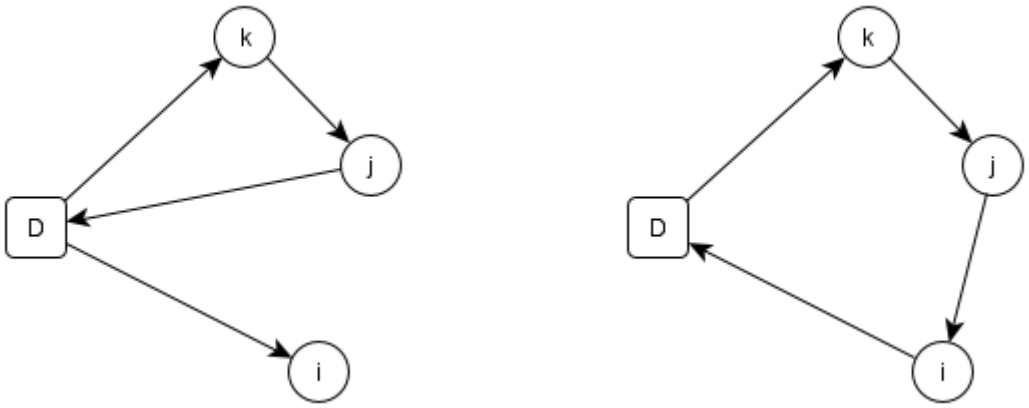


Figura 4 Cálculo del Ahorro Caso 2

Costo Independiente:
$d_{Dk} + d_{kj} + d_{jD} + F * d_{Di}$

Costo Consolidado:
$d_{Dk} + d_{kj} + d_{ji} + d_{iD}$

Total, Ahorro:

$$S_{ij} = d_{Dk} + d_{kj} + d_{jD} + F * d_{Di} - (d_{Dk} + d_{kj} + d_{ji} + d_{iD})$$

$$S_{ij} = d_{jD} + F * d_{Di} - d_{ji} - d_{iD}$$

$$S_{ij} = d_{jD} - d_{ji} + d_{Di} * (F - 1) \qquad (17.2)$$

El procedimiento de inserción de clientes en este paso se presenta más detallada a continuación:

1. Calcular los ahorros a partir de la lógica mostrada en la Figura 4.
2. Ordenar los ahorros en orden descendiente. Empezar en lo más alto de la lista y hacer lo siguiente.
3. Encontrar un enlace factible en la ruta de múltiples clientes que se está analizando con el objetivo de extenderla.
4. Si la ruta no puede seguir siendo extendida, finalizar la ruta.
5. Repetir los pasos (3) y (4) hasta que no haya más enlaces que puedan ser seleccionados.
6. Extraer todas las rutas de múltiples clientes creadas a partir de este procedimiento y los demás clientes que no pudieron ser insertados como rutas subcontratadas de un solo cliente.

17.4 Construcción de rutas subcontratadas

Finalmente se presenta el procedimiento para construir rutas subcontratadas con los clientes restantes. Como se mencionó en las etapas anteriores, los clientes que no fueron asignados a una ruta propia están siendo atendidos de manera individual por una ruta subcontratada como se muestra en la configuración inicial de la Figura 5.

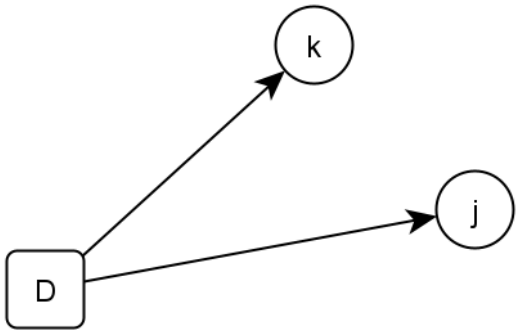


Figura 5 Configuración Inicial – Rutas Subcontratadas

$$\textit{Costo Independiente} = LTL_k + LTL_j = F * d_{Dk} + F * d_{Dj} \qquad (17.3)$$

En esta etapa, como se muestra en la Figura 6, hay dos posibles escenarios para consolidar a dos o más clientes dentro de la misma ruta subcontratada. En el caso 1 se considera la inserción del cliente j a la ruta cuyo nodo final es el cliente k y su ahorro se calcularía con la ecuación $S_{ij} = F * d_{Dj} - F * d_{kj}$ (17.4). En el caso 2 se considera la inserción del cliente k a la ruta cuyo nodo final es el cliente j, su ahorro entonces se calcula con la ecuación $S_{ij} = F * d_{Dk} - F * d_{jk}$ (17.5).

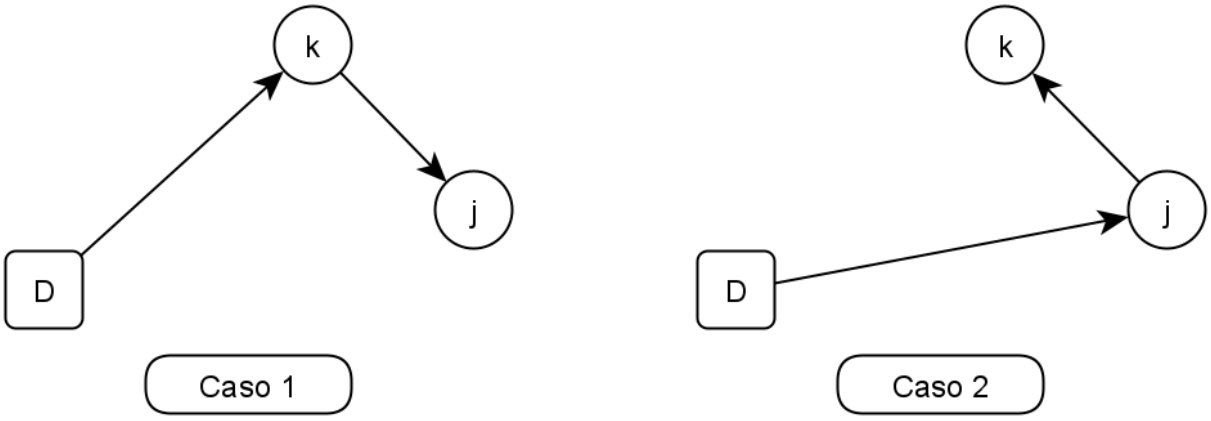


Figura 6 Cálculo del Ahorro

Costo Consolidado:

$$F * d_{Dk} + F * d_{kj}$$

Ahorro:

$$S_{ij} = F * d_{Dk} + F * d_{Dj} - \left(F * d_{Dk} + F * d_{kj}\right)$$

$$S_{ij} = F * d_{Dj} - F * d_{kj} \qquad (17.4)$$

Costo Consolidado:

$$F * d_{Dj} + F * d_{jk}$$

Ahorro:

$$S_{ij} = F * d_{Dk} + F * d_{Dj} - \left(F * d_{Dj} + F * d_{jk}\right)$$

$$S_{ij} = F * d_{Dk} - F * d_{jk} \qquad (17.5)$$

El procedimiento de construcción de rutas subcontratadas en este paso se presenta más detallada a continuación:

1. Calcular los ahorros a partir de la lógica mostrada en la Figura 6, considerado como dos casos diferentes a unión del cliente i con j y la unión del cliente j con i.
2. Ordenar los ahorros en orden descendiente. Empezar en lo más alto de la lista y hacer lo siguiente.
3. Encontrar un enlace factible en la ruta de múltiples clientes que se está analizando con el objetivo de extenderla. En este paso se debe tener en cuenta que los clientes siempre deben ser insertados al final de la ruta y no al principio.
4. Si la ruta no puede seguir siendo extendida, finalizar la ruta.
5. Repetir los pasos (3) y (4) hasta que no haya más enlaces que puedan ser seleccionados.
6. Extraer todas las rutas construidas con este procedimiento.

Al final de este procedimiento se debe tener construido el conjunto de rutas propias de acuerdo con la disponibilidad de la flota y un conjunto de rutas subcontratadas para satisfacer el restante de la demanda. En secciones posteriores de describe la implementación de la metodología del ILS para la etapa de mejoramiento de la solución obtenida en la presente sección.

18 Algoritmo de Búsqueda Local Iterada para la solución del MDVRPPC

En la presente sección se toma como referencia la implementación de la Meta-heurística ILS (Iterated Local Search-Búsqueda Local Iterada) descrita en el capítulo 6 del trabajo de (Subramanian, 2012). No obstante, se implementan nuevos operadores y criterios para el intercambio de clientes no contemplados en dicho trabajo. Así mismo, se mencionan algunas adaptaciones necesarias para el problema multidepósito con flota propia y subcontratada.

Como se indicó en secciones anteriores el ILS funciona perturbando soluciones óptimas locales encontradas previamente, luego mediante una búsqueda local repetida produce nuevas soluciones. En el caso específico del problema MDVRPPC los óptimos locales se construyen a partir de métodos de "clusterización" y de la aplicación del algoritmo de ahorros modificado. Luego se realiza la búsqueda local con el objetivo de mejorar la solución. Finalmente se ejecuta un procedimiento de perturbación para encontrar un nuevo punto inicial y repetir el proceso hasta cumplir con un criterio de aceptación. El objetivo de realizar las perturbaciones es el de escapar de óptimos locales. La eficiencia del algoritmo está íntimamente relacionada con el procedimiento de búsqueda local escogido.

A continuación, se describe de manera más detallada el proceso que se sigue en este trabajo para la aplicación del algoritmo ILS.

1. Construcción de una solución inicial: El procedimiento escogido para la construcción de una solución inicial se describe con mejor detalle en las secciones 15, 16 y 17. Dado que se implementaron varios métodos de "clusterización", dependiendo del método escogido se pueden obtener hasta 7 conjuntos diferentes de soluciones iniciales. Una versión modificada del algoritmo de ahorros es utilizada para la construcción de las rutas.
2. Búsqueda Local: A partir de las soluciones iniciales encontradas en el paso anterior se empieza la búsqueda alrededor de las mismas, es decir, distintos movimientos que se hacen a partir de la solución inicial que mejorar el costo total de las rutas. Para lograr esto se utilizan tres grupos de operadores. El primer grupo compuesto por los llamados operadores inter-depósito. En segundo lugar, se aplican los operadores inter-ruta y finalmente los operadores intra-ruta. Cada movimiento realizado solo es posible si cumple con las condiciones de factibilidad y si mejora el costo total del problema.
3. Perturbación: Tiene por objetivo salir de óptimos locales donde el procedimiento de búsqueda local no puede seguir mejorando la solución. En este paso se ejecutan operadores de perturbación que principalmente intercambian clientes entre rutas o intercambian rutas completas entre depósitos. En este caso no es necesaria la mejora del costo total de las rutas únicamente que se mantengan las condiciones de factibilidad.
4. Criterio de aceptación: Establece un punto de parada para la ejecución del algoritmo. Tiene por objetivo detener el procedimiento de búsqueda y evitar se quede estancado en la misma solución, pese a que se estén aplicando los operadores de perturbación. Los criterios implementados se describen a continuación:
    - Número de iteraciones totales: Corresponde al número total de iteraciones programadas para la ejecución del procedimiento completo.

- Número de iteraciones sin que se mejore la solución: Equivale al número de iteraciones que el algoritmo puede hacer sin que se mejore la solución general.

El algoritmo correspondiente al procedimiento general del ILS se presenta a continuación:

Algoritmo ILS

1: $\boldsymbol{Procedimiento\ ILS}$
2: $\boldsymbol{s_0} \leftarrow \boldsymbol{GenerarSolucionIncial}$;
3: $\boldsymbol{s^*} \leftarrow \boldsymbol{BusquedaLocal}(\boldsymbol{s_0})$;
4: $\boldsymbol{while\ si\ se\ cumple\ criterio\ de\ parada\ hacer}$
5: $\quad \boldsymbol{s'} \leftarrow \boldsymbol{Perturbacion}(\boldsymbol{s^*}, \boldsymbol{historico})$;
6: $\quad \boldsymbol{s^{*\prime}} \leftarrow \boldsymbol{BusquedaLocal}(\boldsymbol{s'})$;
7: $\quad \boldsymbol{s^*} \leftarrow \boldsymbol{CiterioAceptacion}(\boldsymbol{s^*}, \boldsymbol{s^{*'}}, \boldsymbol{historico})$;
8: **end ILS**;

Algoritmo 1 Procedimiento general del ILS

18.1 Búsqueda Local RVND

La búsqueda local es realizada por un procedimiento RVND que selecciona las estructuras de vecindario en un orden aleatorio. Sea $N = \{N1, N2, \dots Nr\}$ el conjunto de estructuras de vecindad. Cuando un vecindario dado del conjunto N falla en mejorar la solución incumbente, el procedimiento RVND selecciona otro vecindario del mismo conjunto para continuar la búsqueda a través del espacio de solución. Sin embargo, como se puede observar en las líneas 9 y 15 del Algoritmo 2 se permite que si se trata de la primera iteración del algoritmo este pueda llegar hasta la etapa de mejoramiento intra-ruta, esto con el fin de no descartar operadores inter-depósito o inter-ruta que fallen en la primera iteración en encontrar soluciones de calidad.

El Algoritmo 2 presenta el procedimiento para la aplicación del RVND. Inicialmente se define la lista con todas las estructuras de vecindario con todos los operadores inter-depósito como se muestra en la línea 4. El ciclo principal del procedimiento está entre las líneas 5 y 33, donde un operador inter-depósito es seleccionado de manera aleatoria. En caso de haber mejora (Línea 9) se guarda la solución y se ejecuta el mejoramiento inter-ruta, en caso contrario se descarta el operador y selecciona manera aleatoria uno nuevo. El inicio del mejoramiento inter-ruta se encuentra en la línea 11 donde se define el conjunto de operadores a utilizar. Luego es seleccionado un operador de manera aleatoria y se comprueba si existe mejora (Línea 15), en caso afirmativo se guarda la solución y se ejecuta el mejoramiento intra-ruta. En caso contrario se descarta el operador y uno nuevo es seleccionado de manera aleatoria. La etapa de mejoramiento intra-ruta inicia en la línea 17 donde se definen todos los operadores a utilizar. Luego es seleccionado de manera aleatoria un operador y se comprueba si existe mejora (Línea 21). En caso de no encontrar una mejor solución se descarta el operador. Este procedimiento se puede representar por medio de un diagrama de flujo como se muestra en la Figura 7.

Algorithm RVND

```
1: Procedure RVND(s)
2: Iter = 0;
3: Update ADSs;                                   // Actualizar Estructuras Auxiliares
4: Initialize the Inter − Depot Neighborhood List (DepotNL);
5: while DepotNL ≠ ∅ do
6:     Iter = Iter + 1;
7:     Choose a neighborhood N^(η) ∈ DepoNL at random;
8:     Find the best neighbor s' of s ∈ N^(η);
9:     if f(s') < f(s) and Iter > 1 then
10:        s ← s';
11:        Initialize the Inter − Route Neighborhood List (RouteNL);
12:        while RouteNL ≠ ∅ do
13:            Choose a neighborhood N'^(η) ∈ RouteNL at random;
14:            Find the best neighbor s' of s ∈ N'^(η);
15:            if f(s') < f(s) and Iter > 1 then
16:                s ← s';
17:                Initialize the Intra − Route Neighborhood List (IntraNL);
18:                while IntraNL ≠ ∅ do
19:                    Choose a neighborhood N''^(η) ∈ IntraNL at random;
20:                    Find the best neighbor s' of s ∈ N''^(η);
21:                    if f(s') < f(s) then
22:                        s ← s';
23:                    else
24:                        Remove N''^(η) from the IntraNL;
25:                return s;
26:            else
27:                Remove N'^(η) from the RouteNL;
28:            Update ADSs;
29:        return s;
30:    else
31:        Remove N^(η) from the DepotNL;
32:    Update DepotNL;
33: return s;
34: end RVND.
```

Algoritmo 2 Procedimiento RVND

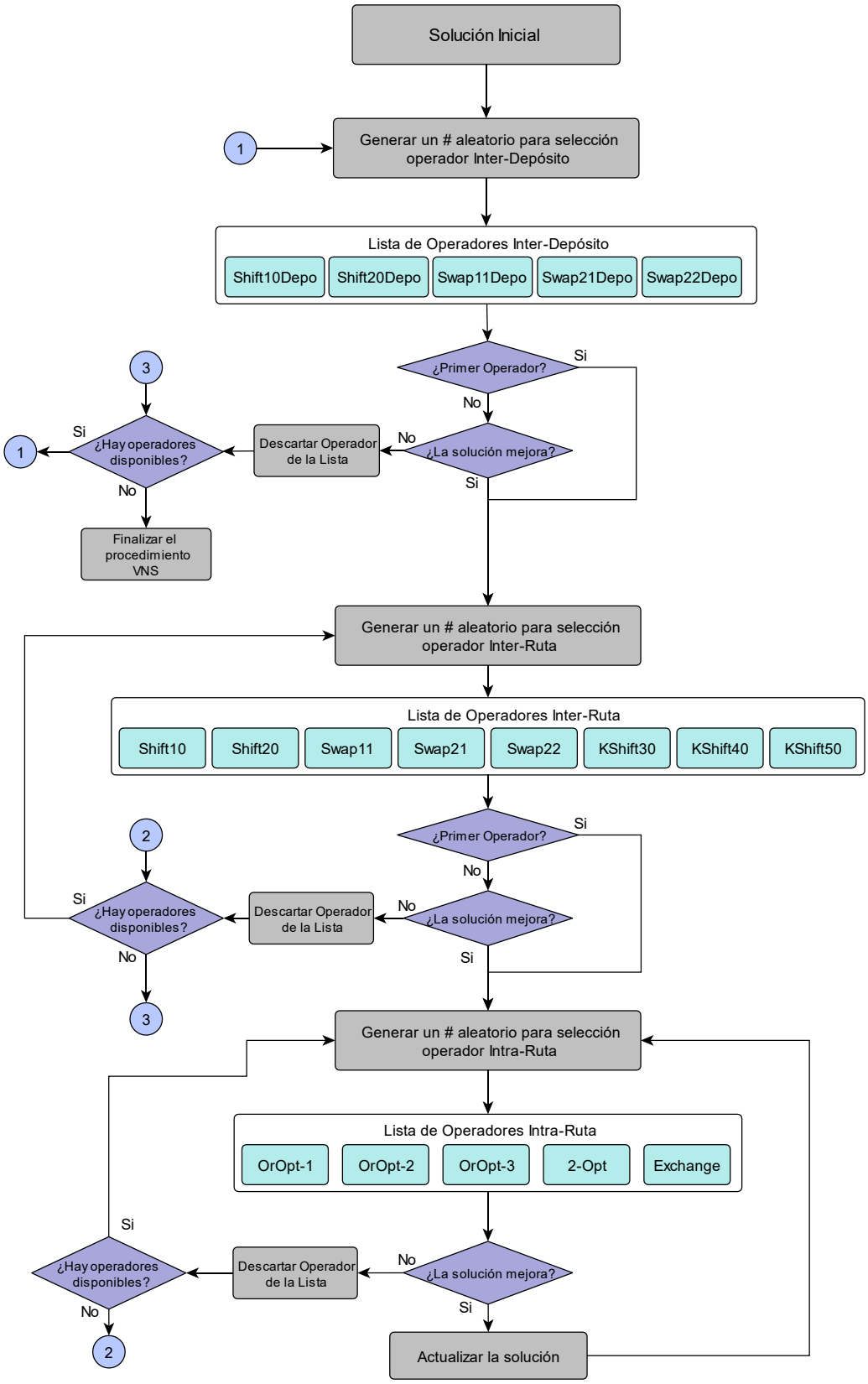


Figura 7 Esquema Random Variable Neighborhood Search

18.2 Heurística ILS-RVND

Ahora se describe el procedimiento ILS que define la generación de la solución inicial, el número máximo de iteraciones y la ejecución de los operadores de perturbación. El procedimiento para ejecutar esta heurística se presenta con más detalle en el Algoritmo 3.

Algorithm ILS-RVND

```
1: Procedure ILS − RVND(MaxIter, MaxIterILS)
2: LoadData( );
3: f* ← ∞;
4: for i: = 1, …, MaxIter do
5:     s ← GenerateInitialSolution(seed); // Solución Inicial Alg. de Ahorros o una variación de esta.
6:     f* ← f(s);
7:     iterILS ← 0;
8:     while iterILS ≤ MaxIterILS do
9:         s ← RVND(s);
10:            if f(s) < f(s') then
11:                s' ← s;
12:                iterILS ← 0;
13:        s ← Perturb(s', seed);
14:        iterILS ← iterILS + 1;
15:    if f(s') < f* then
16:        s* ← s';
17:        f* ← f(s');
18: return s*;
19: end ILS − RVND.
```

Algoritmo 3 Heurística ILS - RVND

Como se puede observar en la línea 4 se genera una solución inicial a la cual se le aplicará el procedimiento de búsqueda local. La solución inicial es generada a partir de la configuración de rutas obtenida con el algoritmo de ahorros modificado o una perturbación de esta. La búsqueda local se realiza con el algoritmo RVND como se muestra en la línea 9. Después, si una mejor solución es encontrada se guarda y se reinicia el contador de $iterILS$. En la línea 13 se tiene la función de perturbación que permite salir de óptimos locales. Después de llegar al límite de iteraciones posibles para el ILS ($MaxIterILS$), se guarda la mejor solución encontrada (Línea 15). El procedimiento se reinicia generando una nueva solución inicial y repitiendo el mismo procedimiento hasta completar la máxima cantidad de iteraciones globales $MaxIter$.

18.3 Estructuras Auxiliares

Con el fin de facilitar la búsqueda local, se adoptaron solo algunas estructuras auxiliares propuestas por (Subramanian et al., 2013), ya que en este trabajo se proponen un conjunto bastante amplio de estructuras auxiliares para tratar principalmente con problemas que incluyen entrega y recogida de bienes. Las estructuras utilizadas en este trabajo son matrices que almacenan información útil sobre cada ruta, a saber:

$SumDelivery$ [ ]: Suma de las demandas de una ruta. Por ejemplo, si $SumDelivery\ [2] = 100$, significa que la suma de las demandas de entrega de todos los clientes de ruta 2 corresponde a 100. La actualización de la estructura $SumDelivery$ [ ]: únicamente consiste en calcular, cada vez que un par de rutas sean modificadas, la suma de la carga anterior de la ruta más la demanda de los clientes que se incluyen a la ruta, menos la demanda de los que salen.

$SumDeliveryDepo$ [ ]: De manera análoga a la anterior se implementa esta estructura para los intercambios entre depósitos. Suma de las demandas de un depósito. Por ejemplo, si $SumDeliveryDepo\ [3] = 250$, significa que la suma de las demandas de los clientes asignados al depósito 3 corresponde a 250. La actualización de la estructura $SumDeliveryDepo$ [ ]: únicamente cosiste en calcular, cada vez que uno o más clientes sean trasladados de un depósito a otro, la suma de los clientes asignados al depósito después de realizar cualquier intercambio.

$NeighborhoodStatus$ [ ] [ ]: Contiene el registro la modificación de una ruta por una estructura de vecindad, es decir, si se realizó un movimiento que mejoró el costo total que implicó la ruta en cuestión. Por ejemplo, si $NeighborhoodStatus\ [1]\ [3] = true$, significa que la última vez que la estructura de vecindad $N^{(1)}$ se aplicó, no se encontró una mejora, movimiento que implicó la ruta 3. Sin embargo, esta ruta fue posteriormente modificada por otra estructura de vecindad o por un movimiento de perturbación. Si $NeighborhoodStatus\ [1]\ [3] = false$, significa que la ruta 3 no sufre ningún cambio después de la última vez que $N^{(1)}$ no tuvo éxito al encontrar un movimiento que implica la mejora de la ruta. Para la actualización de esta estructura, en contraste a la anterior, su complejidad es mayor dado que no solo registra el cambio de una ruta completa, sino también lleva un registro de la aplicación de las estructuras inter-ruta. Así, para cada ruta modificada una verificación es realizada a lo largo de todo el tour para actualizar sus valores.

18.4 Estructuras Inter-Rutas

Se describen los procedimientos correspondientes a los cambios que se hacen entre rutas, es decir, la permutación de uno o más clientes entre recorridos ya existentes. Para el caso múltidepósito estas estructuras solo se ejecutan para la búsqueda local al interior de los depósitos, esto es, únicamente entre rutas pertenecientes al mismo depósito. Estructuras muy similares fueron implementadas para los intercambios entre depósitos y se describen en la sección 18.5. En todas estas estructuras se deben realizar comprobaciones de factibilidad de la ruta, es decir, que la capacidad final de las rutas involucradas en los intercambios no supere la capacidad de los vehículos. Así mismo, como se mencionó en la anterior sección se utiliza la estructura auxiliar $NeighborhoodStatus$, que lleva el registro de los cambios en las rutas, de esta manera se evita volver a ejecutar operadores para un par específico de rutas que se sabe no produce mejoras. También es posible plantear un criterio de distancia para evitar intercambios entre clientes que están muy alejados como se muestra en la Figura 14. Sin embargo, para este trabajo solo se contempló este criterio para los intercambios entre depósitos mencionados en la sección 18.5.

Los operadores sugeridos para las estructuras inter-rutas son los que involucran máximo dos clientes de cada ruta, es decir, que siguen la forma (λ_1, λ_2) tal que $\lambda_1 \leq 2$ y $\lambda_2 \leq 2$. De lo anterior se obtiene que los cambios posibles entre clientes son: $(1,0)$, $(2,0)$, $(1,1)$, $(2,1)$, $(2,2)$. Los intercambios que involucran cambios en un solo sentido son denominados movimientos “Shift”. Por otro lado, los operadores que

implican intercambios de clientes en ambos sentidos son denominados movimientos “Swap”. Estos movimientos deben ser ejecutados de manera exhaustiva probando todos los movimientos posibles y guardando el intercambio que produzca la mejor reducción del costo total. Todas las estructuras de este tipo siguen la misma lógica para ejecutarse en cualquier par de rutas $r1$ y $r2$, esto es que $NeighborhoodStatus\ [\eta][r1] = true$ y $NeighborhoodStatus\ [\eta][r2] = true$. Para variantes del VRP como lo es el MDVRPPC la comprobación de la factibilidad de las rutas es un cálculo trivial que solo implica la suma total de la demanda de la ruta. Lo anterior se realiza incorporando la estructura auxiliar $SumDelivery$ [ ] por lo que no implica un aumento significativo en la complejidad computacional.

A continuación, se describen los operadores implementados:

$Shift(1,0)$: Este operador toma un cliente k asignado a una ruta $r1$ y lo traslada a otra ruta $r2$ perteneciente al mismo deposito. Para cumplir las condiciones de factibilidad de la ruta se comprueba si $dk + SumDelivery[r2] \leq Q$, obviamente no es necesario comprobar la factibilidad de la ruta $r1$ dado que la carga asignada a esta no está aumentando. El procedimiento para realizar este tipo de intercambios se muestra en el Algoritmo 4.

Algorithm Shift(1,0)

```
1: Procedure Shift (1,0)(s)
2: for r1 = 1 ... v do
3:    for r2 = 1 ... v do
4:       if r1 ≠ r2 and (NeighborhoodStatus[1][r1 ] = true or NeighborhoodStatus[1][r2 ] =
   true) then
5:          for every customer k ∈ r1 do
6:             if d_k + SumDelivery[r2] ≤ Q then
7:                for each position z in r2 do
8:                   Evaluate the solution cost f (s' ) of the neighborhood solution s' of s, i.e., the cost of
                           transfering k ∈ r1 to the position z in r2;
9:                   if f (s' ) < f* then
10:                     if s' is feasible then
11:                        f* ← f (s' );
12:                        s* ← s';
13: if f* < f (s) then
14:    s ← s*;
15: else
16:    Update NeighborhoodStatus;
17: return s;
18: end Shift − (1,0).
```

Algoritmo 4 Operador Shift (1,0)

Para mostrar el funcionamiento de este operador se presenta la Figura 8. A manera de ejemplo, se tiene el intercambio del nodo 8 de la ruta $r1 = [8\ 9\ 10\ 11]$ y que será movido a la ruta $r2 = [4\ 5\ 6\ 7]$. Después de realizado el intercambio las rutas quedan de la siguiente forma: $r1 = [9\ 10\ 11]$ y $r2 = [4\ 5\ 6\ 8\ 7]$.

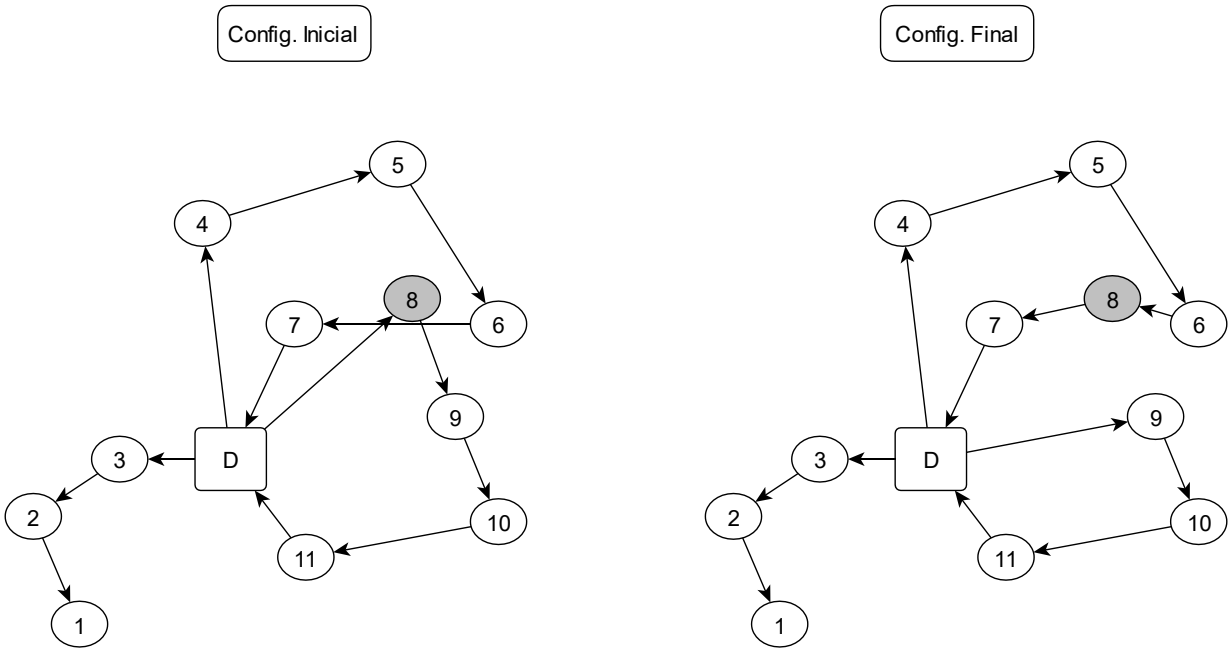


Figura 8 Shif(1,0)

$Shift(2,0)$: Este operador toma dos clientes adyacentes k y l asignados a la ruta $r1$ y los traslada a la ruta $r2$ perteneciente al mismo deposito. Para cumplir las condiciones de factibilidad de la ruta se comprueba si $dk + dl + SumDelivery[r2] \leq Q$, en este caso tampoco no es necesario comprobar la factibilidad de la ruta $r1$ dado que la carga asignada a esta no está aumentando. El procedimiento para realizar este tipo de intercambios se muestra en el Algoritmo 5.

Algorithm Shift(2,0)

1: *Procedure Shift* (2,0) (s)
2: ***for*** $r_1 = 1 \ldots v$ ***do***
3: ***for*** $r_2 = 1 \ldots v$ ***do***
4: ***if*** $r1 \neq r2$ ***and*** ($NeighborhoodStatus[3][r1] = true$ ***or*** $NeighborhoodStatus[3][r2] = true$) ***then***
5: ***for*** *every pair of adjacent customers k and* $l \in r_1$ ***do***
6: ***if*** $d_k + d_l + SumDelivery[r2] \leq Q$ ***then***
7: ***for*** *each position z in* $r2$ ***do***
8: *Evaluate the solution cost* $f(s')$ *of the neighborhood solution* s' *of* s, *i.e., the cost of transfering k and* $l \in r_1$ *to the position z in* $r2$;
9: ***if*** $f(s') < f^*$ ***then***
10: ***if*** s' *is feasible* ***then***
11: $f^* \leftarrow f(s')$;
12: $s^* \leftarrow s'$;
13: ***if*** $f * < f(s)$ ***then***
14: $s \leftarrow s^*$;
15: ***else***
16: *Update NeighborhoodStatus*;
17: ***return*** s;

18: $end\ Shift - (2,0)$.

Algoritmo 5 Operador Shift (2,0)

Para mostrar el funcionamiento de este operador se presenta la Figura 9. A manera de ejemplo, se tiene el intercambio de los nodos 8 y 9 de la ruta $r1 = [8\ 9\ 10\ 11]$ y que serán movidos a la ruta $r2 = [4\ 5\ 6\ 7]$. Después de realizado el intercambio las rutas quedan de la siguiente forma: $r1 = [10\ 11]$ y $r2 = [4\ 5\ 6\ 8\ 9\ 7]$.

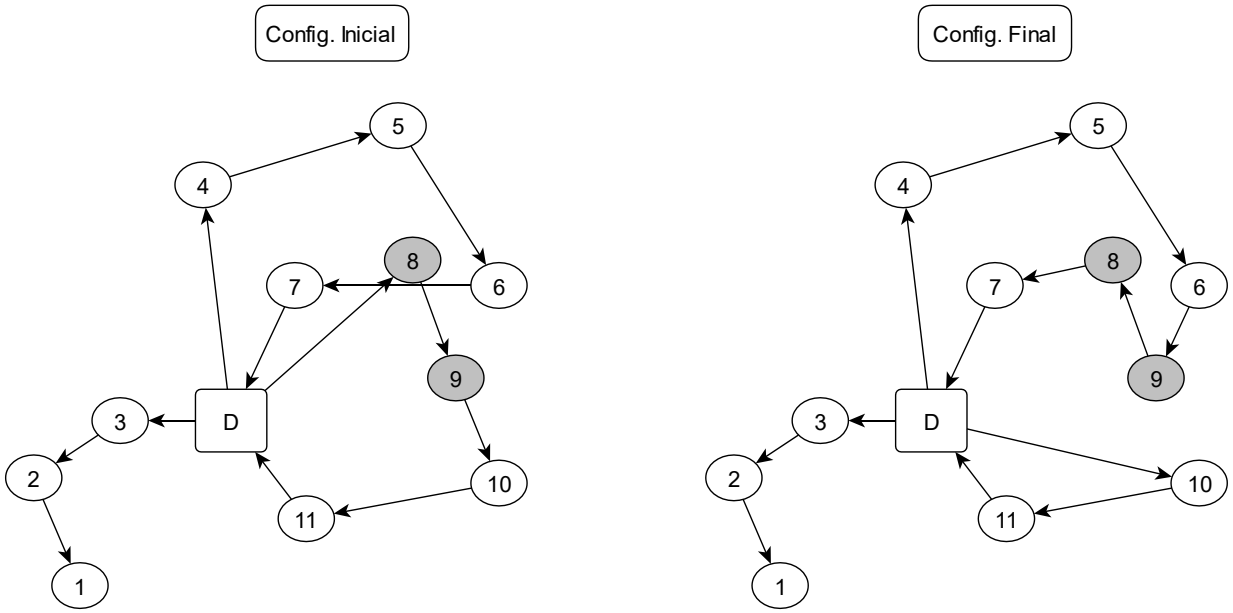


Figura 9 Shif(2,0)

$Swap(1,1)$: Este operador toma un cliente k asignado a la ruta $r1$ y lo intercambia con el cliente l perteneciente a la ruta $r2$ del mismo depósito. Para cumplir las condiciones de factibilidad de la ruta se comprueba si $dk + SumDelivery[r2] - dl \leq Q$ y $dl + SumDelivery[r1] - dk \leq Q$. En este caso es necesario comprobar la factibilidad de ambas rutas dado que la carga asignada a cada una puede sobrepasar de manera independiente la capacidad de los vehículos. El procedimiento para realizar este tipo de intercambios se muestra en el Algoritmo 6.

Algorithm Swap(1,1)

1: $Procedure\ Swap\ (1,1)\ (s)$
2: $\boldsymbol{for}\ r_1 = 1 \ldots v\ \boldsymbol{do}$
3: $\quad \boldsymbol{for}\ r_2 \neq r_1 + 1 \ldots v\ \boldsymbol{do}$
4: $\quad\quad \boldsymbol{if}\ (NeighborhoodStatus[2][r_1] = true\ \boldsymbol{or}\ NeighborhoodStatus[2][r_2] = true)\ \boldsymbol{then}$
5: $\quad\quad\quad \boldsymbol{for}\ every\ customer\ k \in r_1\ \boldsymbol{do}$
6: $\quad\quad\quad\quad \boldsymbol{for}\ every\ customer\ l \in r_2\ \boldsymbol{do}$
7: $\quad\quad\quad\quad\quad \boldsymbol{if}\ d_k + SumDelivery[r2] - dl \leq Q\ \boldsymbol{and}\ d_l + SumDelivery[r2] - dk \leq Q\ \boldsymbol{then}$

8: *Evaluate the solution cost* $f(s')$ *of the neighborhood solution* s' *of* s, *i.e., the cost of exchanging* $k \in r_1$ *with* l *in* r_2;
9: **if** $f(s') < f *$ **then**
10: **if** s' *is feasible* **then**
11: $f^* \leftarrow f(s')$;
12: $s^* \leftarrow s$;
13: **if** $f^* < f(s)$ **then**
14: $s \leftarrow s^*$;
15: **else**
16: *Update NeighborhoodStatus*;
17: **return** s;
18: *end* $Swap - (1,1)$.

Algoritmo 6 Operador Swap (1,1)

Para mostrar el funcionamiento de este operador se presenta la Figura 10. A manera de ejemplo, se tiene el intercambio de los nodos 8 de la ruta $r1 = [8\ 9\ 10\ 11]$ y 6 perteneciente a la ruta $r2 = [4\ 5\ 6\ 7]$. Después de realizado el intercambio las rutas quedan de la siguiente forma: $r1 = [6\ 9\ 10\ 11]$ y $r2 = [4\ 5\ 8\ 7]$.

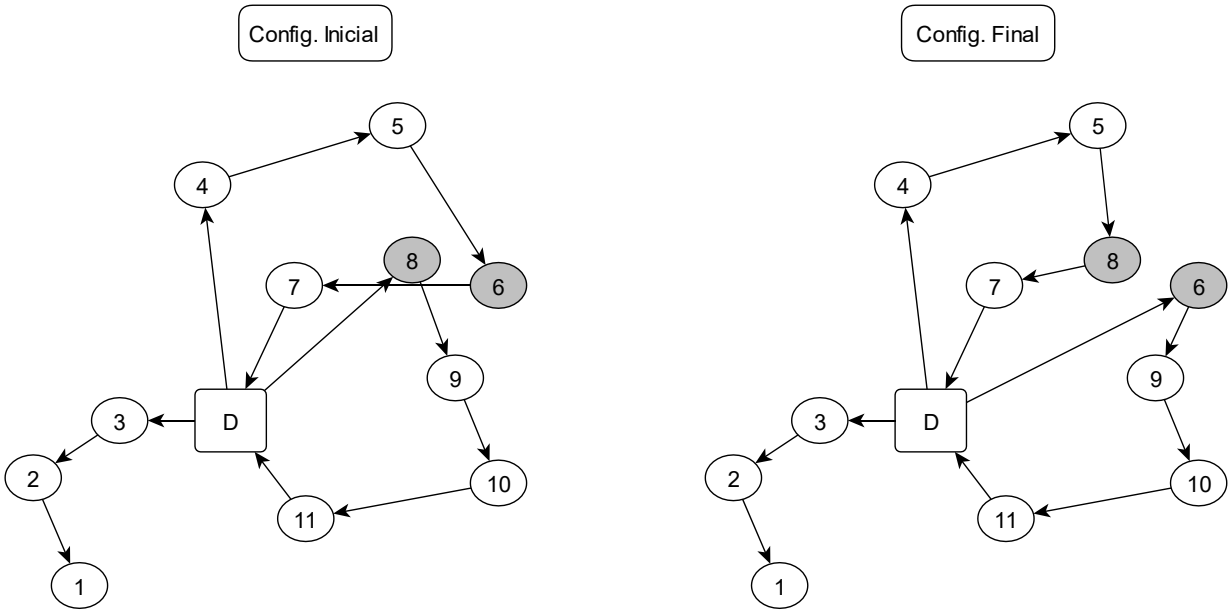


Figura 10 Swap(1,1)

$Swap(2,1)$: Este operador toma dos clientes adyacentes k y l asignados a la ruta $r1$ y los intercambia con el cliente k' perteneciente a la ruta $r2$ del mismo depósito. Para cumplir las condiciones de factibilidad de la ruta se comprueba si $dk + dl + SumDelivery[r2] - dk' \leq Q$ y $dk' + SumDelivery[r1] - dk - dl \leq Q$. En este caso es necesario comprobar la factibilidad de ambas rutas dado que la carga

asignada a cada una puede sobrepasar de manera independiente la capacidad de los vehículos. El procedimiento para realizar este tipo de intercambios se muestra en el Algoritmo 7.

Algorithm Swap(2,1)

1: *Procedure Swap* (2,1) (s)
2: ***for*** $r_1 = 1 \dots v$ ***do***
3: ***for*** $r_2 = 1 \dots v$ ***do***
4: ***if*** $r1 \neq r2$ ***and*** $(NeighborhoodStatus[4][r_1] = true$ ***or*** $NeighborhoodStatus[4][r_2] = true)$ ***then***
5: ***for*** *every pair of adjacent customers* k *and* $l \in r_1$ ***do***
6: ***for*** *every customer* k' *in* r_2 ***do***
7: ***if*** $d_k + d_l + SumDelivery[r_2] - dk' \leq Q$ ***and*** $d_{k'} + SumDelivery[r_1] - dk - dl \leq Q$ ***then***
8: *Evaluate the solution cost* $f(s')$ *of the neighborhood solution* s' *of* s, *i.e., the cost of exchanging adjacent customers* k *and* $l \in r_1$ *with* $k' \in r_2$;
9: ***if*** $f(s') < f^*$ ***then***
10: ***if*** s' *is feasible* ***then***
11: $f^* \leftarrow f(s')$;
12: $s^* \leftarrow s'$;
13: ***if*** $f^* < f(s)$ ***then***
14: $s \leftarrow s^*$;
15: ***else***
16: *Update NeighborhoodStatus*;
17: ***return*** s;
18: *end Swap* $-$ (2,1).

Algoritmo 7 Operador Swap (2,1)

Para mostrar el funcionamiento de este operador se presenta la Figura 11. A manera de ejemplo, se tiene el intercambio del nodo 8 de la ruta $r1 = [8\ 9\ 10\ 11]$ y los nodos 6 y 7 pertenecientes a la ruta $r2 = [4\ 5\ 6\ 7]$. Después de realizado el intercambio las rutas quedan de la siguiente forma: $r1 = [7\ 6\ 9\ 10\ 11]$ y $r2 = [4\ 5\ 8]$.

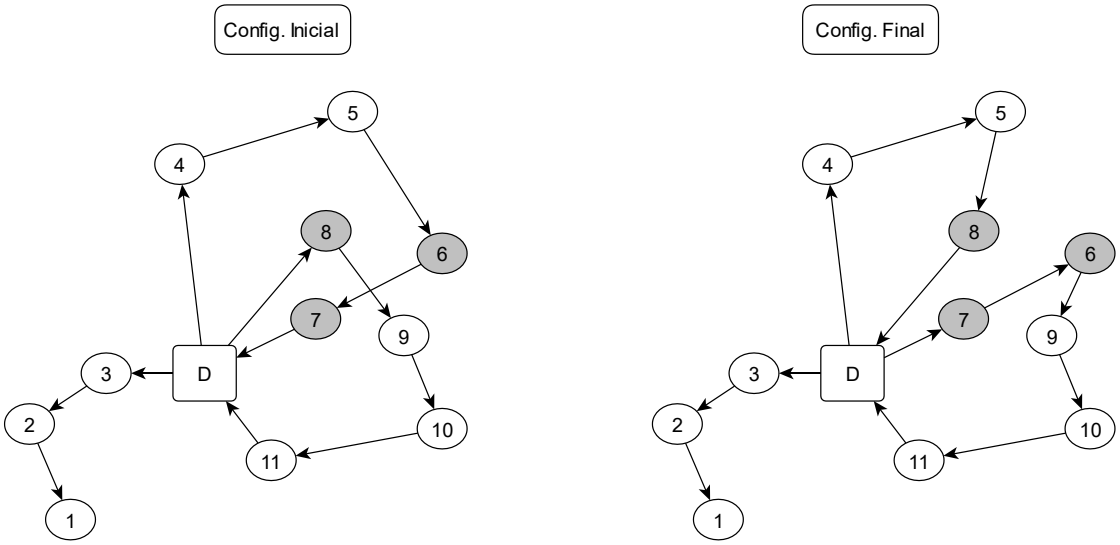


Figura 11 Swap(2,1)

$Swap(2,2)$: Este operador toma dos clientes adyacentes k y l asignados a la ruta $r1$ y los intercambia con los clientes k' y l' pertenecientes a la ruta $r2$ del mismo depósito. Para cumplir las condiciones de factibilidad de la ruta se comprueba si $dk + dl + SumDelivery[r2] - dk' - dl' \leq Q$ y $dk' + dl' + SumDelivery[r1] - dk - dl \leq Q$. En este caso es necesario comprobar la factibilidad de ambas rutas dado que la carga asignada a cada una puede sobrepasar de manera independiente la capacidad de los vehículos. El procedimiento para realizar este tipo de intercambios se muestra en el Algoritmo 8.

Algorithm Swap(2,2)

```
1: Procedure Swap − (2,2)(s)
2: for r1 = 1 ... v do
3:    for r2 = r1 + 1 ... v do
4:       if r1 ≠ r2 and (NeighborhoodStatus[5][r1] = true or
               NeighborhoodStatus[5][r2] = true) then
5:          for every pair of adjacent customers k and l ∈ r1 do
6:             for every pair of adjacent customers k' and l' ∈ r2 do
7:                if dk + dl + SumDelivery[r2] − dk' − dl' ≤ Q and
                     dk' + dl' + SumDelivery[r1] − dk − dl ≤ Q ≤ Q then
8:                   Evaluate the solution cost f(s')of the neighborhood solution s' of s, i.e., the
                        cost of exchanging adjacent customers k and l ∈ r1 with
                              adjacent customers k' and k' ∈ r2;
9:                if f (s' ) < f ' then
10:                  if s' is feasible then
11:                     f* ← f (s' );
12:                     s* ← s' ;
13: if f* < f (s) then
14:    s ← s*;
15: else
16:    Update NeighborhoodStatus;
17: return s;
18: end Swap − (2,2).
```

Algoritmo 8 Operador Swap (2,2)

Para mostrar el funcionamiento de este operador se presenta la Figura 12. A manera de ejemplo, se tiene el intercambio de los nodos 8 y 9 de la ruta $r1 = [8\ 9\ 10\ 11]$ y los nodos 6 y 7 pertenecientes a la ruta $r2 = [4\ 5\ 6\ 7]$. Después de realizado el intercambio las rutas quedan de la siguiente forma: $r1 = [7\ 6\ 10\ 11]$ y $r2 = [4\ 5\ 9\ 8]$.

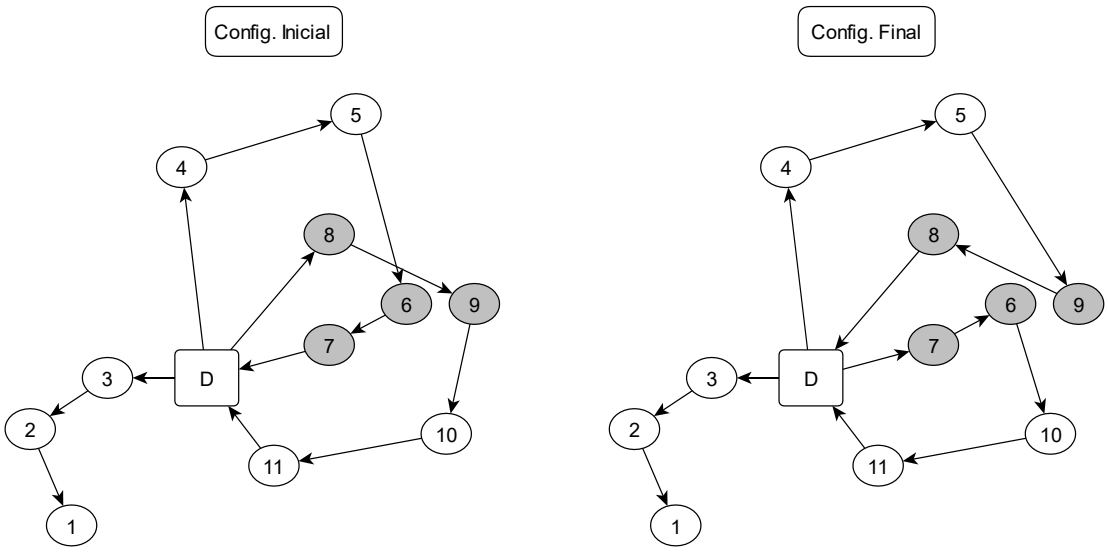


Figura 12 Swap(2,2)

$K - Shift$: Este operador toma k clientes adyacentes asignados a la ruta $r1$ y los traslada al final de la ruta $r2$ perteneciente al mismo deposito. Para cumplir las condiciones de factibilidad de la ruta se comprueba si $\sum di \ + \ SumDelivery[r2] \leq Q \ \ con\, i = 1,2, \dots k,$ en este caso no es necesario comprobar la factibilidad de la ruta $r1$ dado que la carga asignada a esta no está aumentando. Este operador es equivalente a generar un subconjunto de intercambios contenidos en el conjunto generado por el operador $Shift(k, 0)$, por lo que para este trabajo solo se contemplan valores de $k \geq 3$. El procedimiento para realizar este tipo de intercambios se muestra en el Algoritmo 9.

Algorithm K-Shift

1: $Procedure\ K\ -Shift(s)$
2: ***for*** $r_1 \ = \ 1 \dots v$ ***do***
3: ***for*** $r_2 \ = \ 1 \dots v$ ***do***
4: ***if*** $r1 \neq r2$ ***and*** $Q_{r1} \ < \ Q_{r2}$ ***and*** $(NeighborhoodStatus[6][r_1] \ = \ true \quad \textbf{or} \quad NeighborhoodStatus[6][r_2] \ = \ true)$ ***then***
5: ***for*** *every k adjacent customers* $\in r_1$ ***do***
6: ***if*** $\sum_1^k d_i \ + \ SumDelivery[r_2] \ \leq Q_{r2}$ ***then***
7: $Evaluate\ the\ solution\ cost\ f\ (s')of\ the\ neighborhood\ solution\ s'$ $of\ s, i.e., the\ cost\ of transfering\ the\ k\ adjacent$ $customers \ \in \ r_1 \ to\ the\ end\ of\ r_2;$
8: ***if*** $f\ (s') \ < \ f^*$ ***then***
9: ***if*** s' $is\ feasible$ ***then***
10: $f^* \ \leftarrow \ f\ (s')$;
11: $s^* \leftarrow s'$;
11: ***if*** $f^* \ < \ f\ (s)$ ***then***
12: $s \leftarrow s^*$;
13: ***else***
14: $Update\ NeighborhoodStatus$;
15: ***return*** s;
16: $end\ K - Shift.$

Algoritmo 9 Operador K-Shift

Para mostrar el funcionamiento de este operador se presenta la *Figura 13 KShif(k,0)*Figura 13. A manera de ejemplo, se tiene el intercambio de los nodos entre dos rutas para los valores $k = 1, k = 2\ y\ k = 3$. Este tipo de no suelen ser factibles para valores grandes de k ya que fácilmente se puede sobrepasar la capacidad de los vehículos.

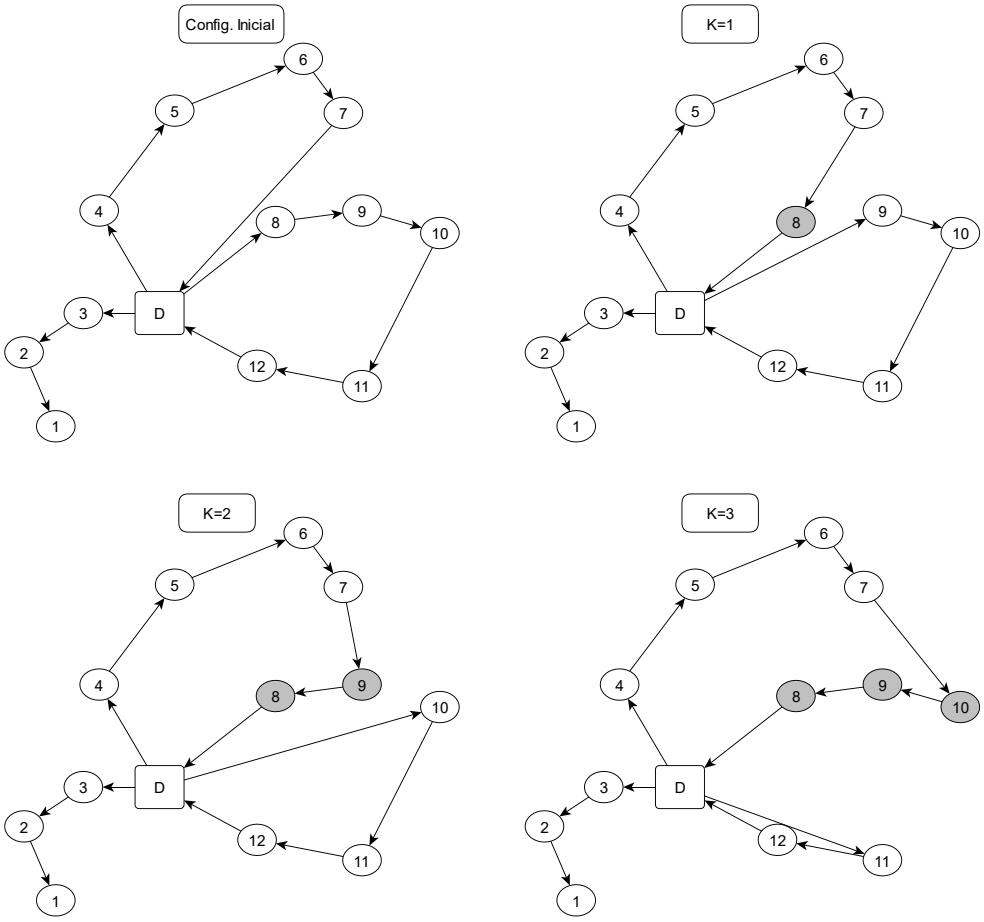


Figura 13 KShif(k,0)

18.5 Estructuras Inter-Depósito

En esta sección se describen los operadores implementados para realizar cambios de clientes y rutas entre depósitos. Después de varias pruebas se decidió implementar operadores muy similares a los descritos en la sección 18.4. De esta manera se implementó una versión modificada de estos operadores llamados $Shift(1,0)Depo$, $Shift(2,0)Depo$, $Swap(1,1)Depo$, $Swap(2,1)Depo$ y $Swap(2,2)Depo$ que básicamente consiste realizar el mismo proceso de intercambio de clientes pero considerando rutas de diferentes depósitos y he implementado un criterio de distancia. Por lo tanto, los clientes que pueden ser traslados de una ruta a la otra no deben exceder la distancia preestablecida como se muestra en la Figura 14.

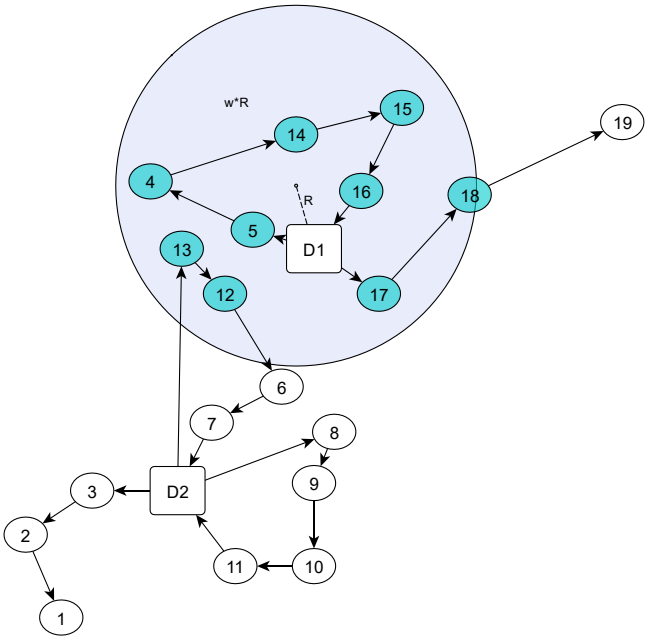


Figura 14 Lógica de Distancia

Como se muestra en la Figura 14 los clientes que están ubicados a *w* veces la distancia entre el centroide de las rutas y su depósito, son clientes que son susceptibles a ser intercambiados. En este caso únicamente los clientes 13 y 12 pertenecientes al depósito D2 son susceptibles a ser intercambiados con respecto a la ruta del depósito D1.

Basándose en las consideraciones hechas arriba, a continuación, se presentan los operadores desarrollados para las estructuras Inter depósito. Es importante aclarar que ahora no solamente es necesario probar la factibilidad de las rutas (garantizar que no se supere la capacidad de los vehículos), sino que también se debe probar la factibilidad del depósito, es decir, que la demanda de los clientes asignados a un depósito no supere la capacidad del depósito.

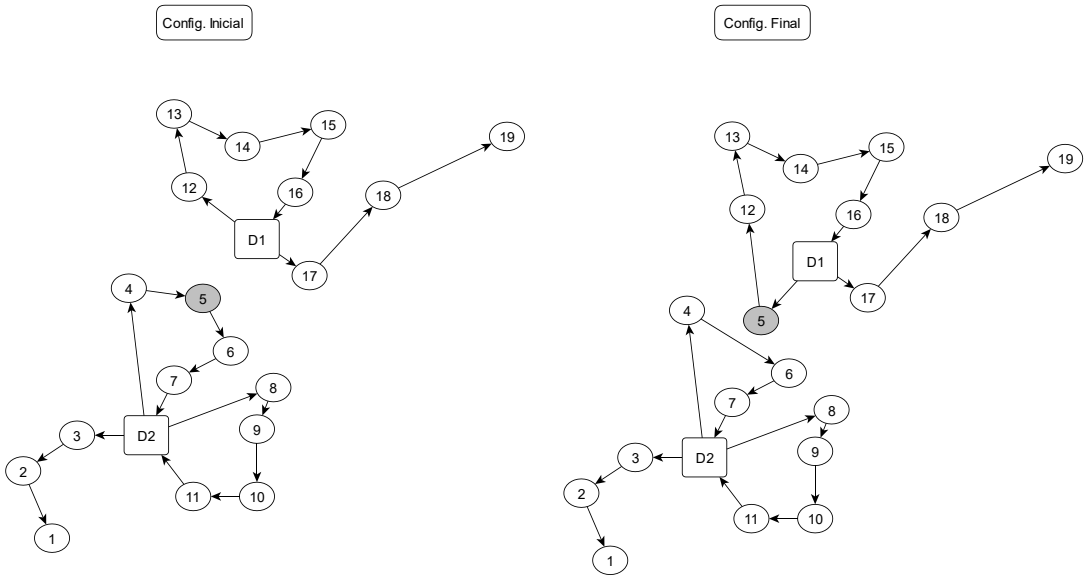


Figura 15 Shif(1,0)Depo

En la Figura 15 se muestra el funcionamiento del operador $Shift(1,0)Depo$, este operador toma un nodo asignado a una ruta de un depósito y lo traslada a una ruta perteneciente a un depósito diferente. Se deben cumplir las condiciones de factibilidad de la ruta, factibilidad del depósito y el criterio de distancia previamente descrito. Para el ejemplo se muestra el intercambio del nodo 5 perteneciente al depósito D2 y se traslada a una ruta del depósito D1.

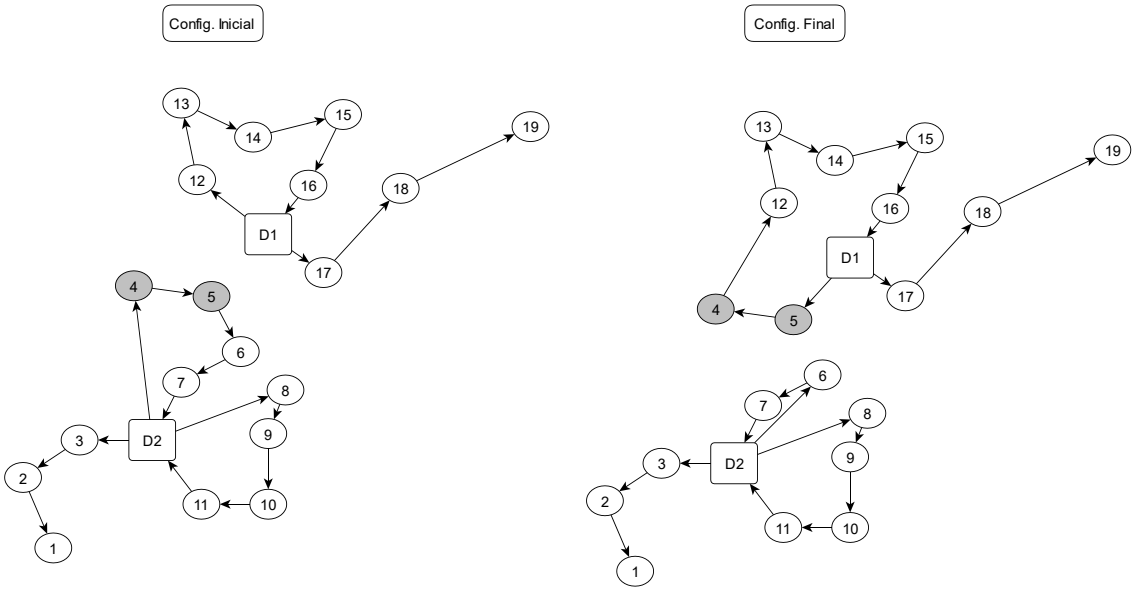


Figura 16 Shif(2,0)Depo

En la Figura 16 se muestra el funcionamiento del operador $Shift(2,0)Depo$, este operador toma dos nodos asignados a una ruta de un depósito y los traslada a una ruta perteneciente a un depósito diferente. Se deben cumplir las condiciones de factibilidad de la ruta, factibilidad del depósito y el criterio de distancia previamente descrito. Para el ejemplo se muestra el intercambio de los nodos 4 y 5 pertenecientes al depósito D2 y se trasladan a una ruta del depósito D1.

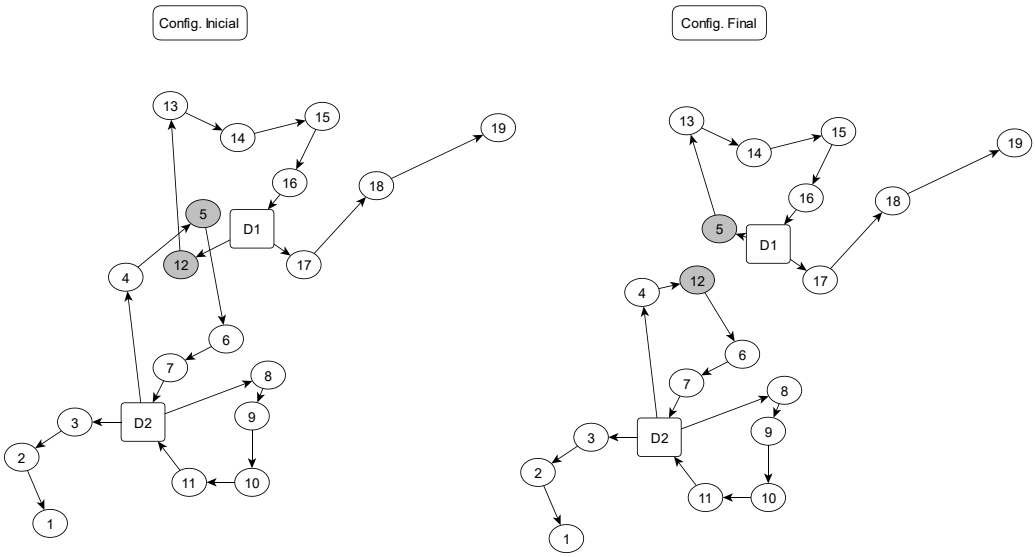


Figura 17 Swap(1,1)Depo

En la Figura 17 se muestra el funcionamiento del operador $Swap(1,1)Depo$, este operador toma dos nodos asignados a dos rutas pertenecientes a diferentes depósitos y los intercambia entre sí. Se deben cumplir las condiciones de factibilidad de la ruta, factibilidad del depósito y el criterio de distancia previamente descrito. Para el ejemplo se muestra el intercambio de los nodos 12 y 5 pertenecientes a rutas en los depósitos D2 y D1 respectivamente.

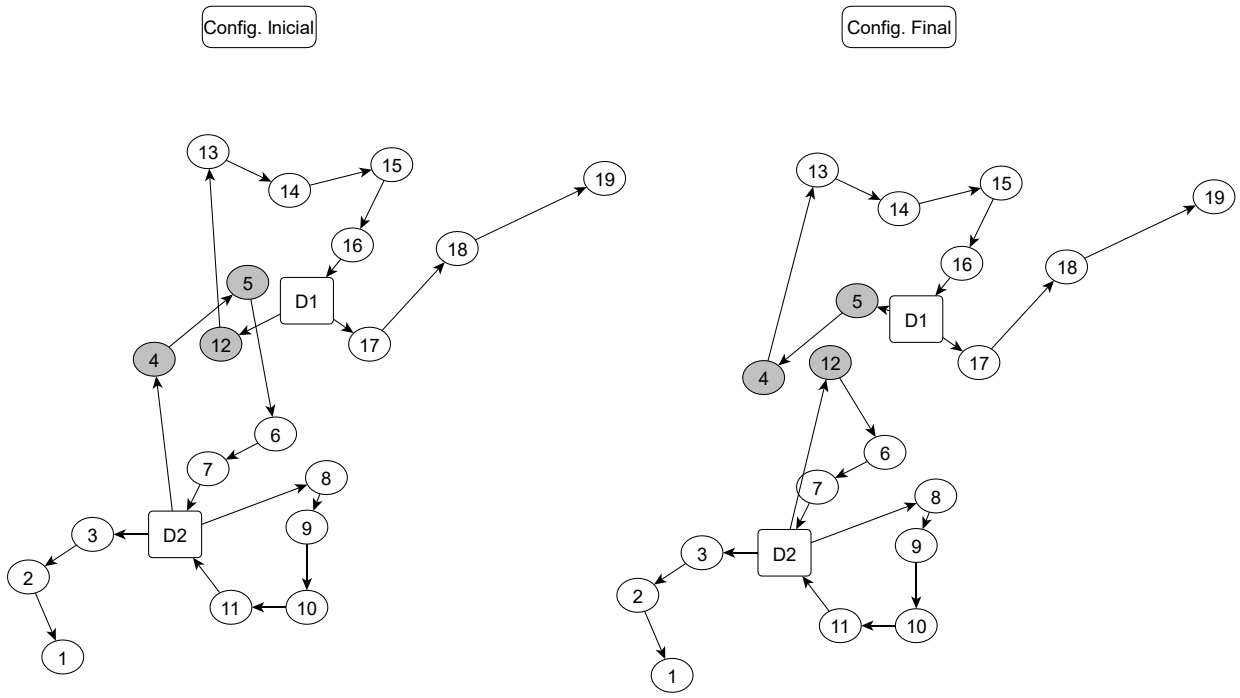


Figura 18 Swap(2,1)Depo

En la Figura 18 se muestra el funcionamiento del operador $Swap(2,1)Depo$, este operador toma dos nodos asignados a una ruta de un depósito específico y los intercambia con un nodo de una ruta pertenecientes a un depósito diferente. Se deben cumplir las condiciones de factibilidad de la ruta, factibilidad del depósito y el criterio de distancia previamente descrito. Para el ejemplo se muestra el intercambio de los nodos 4 y 5 pertenecientes a ruta en el depósito D2 con el nodo 12 del depósito D1.

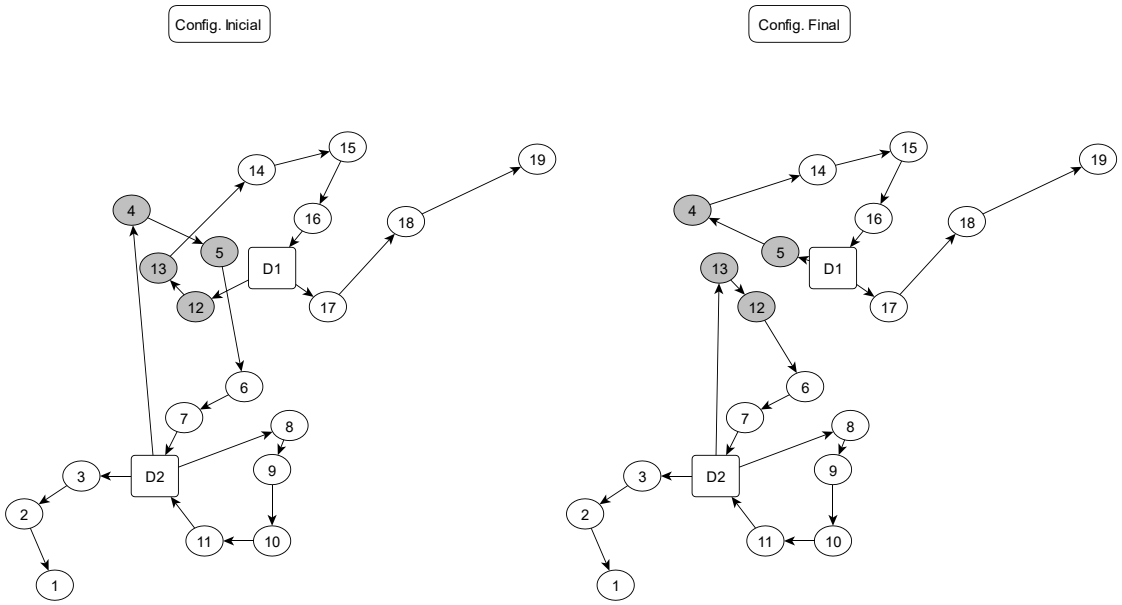


Figura 19 Swap(2,2)Depo

En la Figura 19 se muestra el funcionamiento del operador $Swap(2,2)Depo$, este operador toma cuatro nodos, dos asignados a una ruta de un depósito y los otros dos a una ruta perteneciente a un depósito diferente y los intercambia entre sí. Se deben cumplir las condiciones de factibilidad de la ruta, factibilidad del depósito y el criterio de distancia previamente descrito. Para el ejemplo se muestra el intercambio de los nodos 4 y 5 pertenecientes a ruta en el depósito D2 con los nodos 12 y 13 del depósito D1.

Finalmente cabe mencionar que los operadores que hacen intercambios de rutas completas entre depósitos no se tomaron en cuenta para este trabajo. Lo anterior es debido a que es poco probable mejorar el costo total al hacer un intercambio completo de una o más rutas entre depósitos. No obstante, bajo este mismo criterio se implementaron operadores similares en la etapa de perturbación del modelo ya que no es necesario que se mejore la solución después de realizar una perturbación.

18.6 Estructuras Intra-Ruta

En la presente sección se describen los operadores intra-ruta que se encargan del mejoramiento de las rutas mediante el reposicionamiento de sus clientes. Al Igual que los operadores utilizados en los intercambios inter-depósito e inter-ruta, estos operadores deben ser aplicados de manera exhaustiva probando todas las posibles combinaciones según la lógica de cada uno. En esta fase de mejoramiento no es necesaria la comprobación de factibilidad dado que este tipo de intercambios no cambian la carga asignada a los vehículos ni a los depósitos.

Los 5 operadores implementados en el presente trabajo se muestran en la Figura 20.

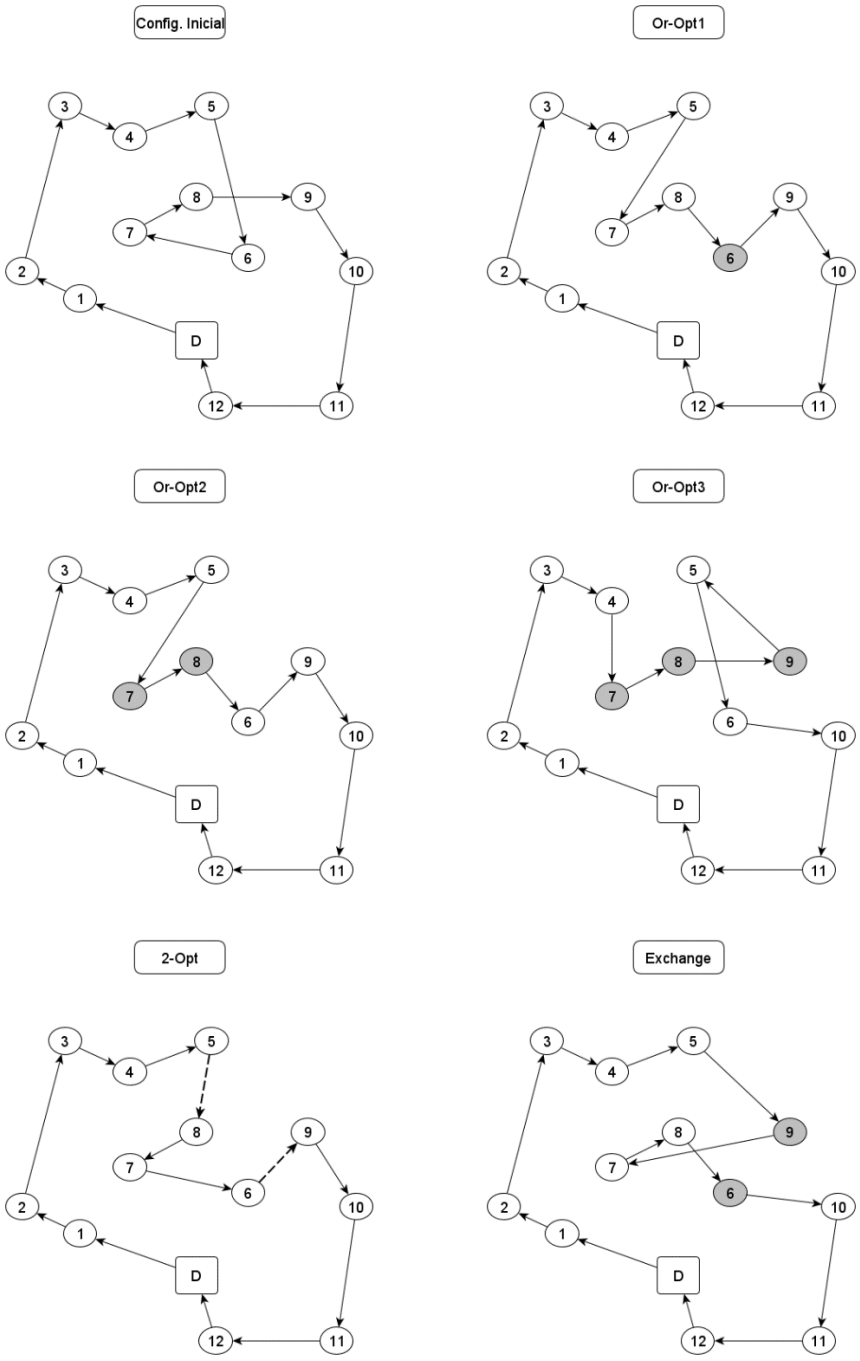


Figura 20 Operadores Intra Ruta

$2Opt$: Consiste en invertir el orden de un segmento de la ruta, lo que es equivalente a remover dos arcos no adyacentes e intercambiarlos de posición. En la Figura 20 se muestra como el arco entre los clientes $(6,7)$ y el arco entre los clientes $(8,9)$ son intercambiados, es decir, el orden de los clientes que inicialmente era $[\ldots 5\ 6\ 7\ 8\ 9\ \ldots]$ queda como $[\ldots 5\ 8\ 7\ 6\ 9\ \ldots]$.

$Or - Opt1$: También llamado reinserción, consiste en quitar un nodo y reinsertarlo en una posición diferente de la ruta. Para el ejemplo en la Figura 20 se cambia de posición el cliente 6.

$Or - Opt2$: Bajo el mismo principio del anterior, este operador realizar el cambio de posición de dos nodos adyacentes dentro de la misma ruta. En la Figura 20 se muestra el intercambio de los nodos 6 y 7 son cambiados de posición.

$Or - Opt3$: Este operador de manera similar remueve tres clientes adyacentes de la ruta y los reinserta en otra posición. Para el ejemplo en la Figura 20 se cambian de posición los clientes 7,8 y 9.

$Exchange$: Este operador hace un intercambio entre dos nodos pertenecientes a la misma ruta. En la Figura 20 se muestra el intercambio de los clientes 6 y 9 por lo que la configuración final como $[\dots\ 5\ 9\ 7\ 8\ 6\ 10\ \dots]$.

18.7 Perturbación

En esta sección se describe la implementación de los operadores utilizados para la perturbación. Después de terminar la búsqueda local de cualquier iteración, se ejecuta de manera aleatoria uno de estos operadores de perturbación. Todos ellos tienen como propósito distorsionar la mejor solución encontrada hasta el momento y diversificar el espacio de búsqueda. También es importante mencionar que los movimientos hechos en esta etapa no requieren que la solución mejore.

$MultiSwap(1,1)$: Realiza múltiples cambios aleatorios de clientes entre dos rutas a la vez. En este trabajo se definió el número de cambios como $0.5 * v$, donde v es el número de vehículos en la solución. Lo anterior significa que se escoge un par de rutas de manera aleatoria $0.5 * v$ veces, y en cada una se intercambian de manera aleatoria dos clientes. Dado que este operador puede intercambiar clientes de rutas no solamente del mismo depósito, sino que también entre rutas de diferentes depósitos, el criterio de distancias presentado en la Figura 14 es también utilizado para evitar intercambios con clientes muy alejados.

$MultiShift(1,1)$: Realiza múltiples cambios aleatorios de clientes entre dos rutas a la vez. En este operador el número de cambios es seleccionado de manera aleatoria a partir de la formula $n°_{Cambios} = F_{Cambio} * v$, donde F_{Cambio} es un número aleatorio distribuido uniformemente en el intervalo de $[0.5, 1.5]$ y v es el número de vehículos en la solución. Lo anterior significa que se escoge un par de rutas de manera aleatoria $n°_{Cambios}$ veces, y en cada una se intercambian de manera aleatoria dos clientes. Dado que este operador puede intercambiar clientes de rutas no solamente del mismo depósito, sino que también entre rutas de diferentes depósitos, el criterio de distancias presentado en la Figura 14 es también utilizado para evitar intercambios con clientes muy alejados.

Los siguientes operadores de perturbación son utilizados para realizar cambios de rutas completas entre depósitos. Estos operadores no se incluyeron en la sección de estructuras inter-depósito ya que para la mayoría de los cambios de este tipo la solución no mejora. Por otro lado, los operadores inter-ruta e intra-

ruta pueden reorganizar una configuración de clientes asignados a un depósito que parece no ser óptima, permitiendo una mejor exploración del espacio de soluciones.

$ShiftRutaD$: Este operador remueve una ruta completa perteneciente a un depósito y la asigna a otro. El criterio para hacer movimientos de este tipo es que la solución obtenida sea factible, es decir, que no se supera la capacidad del depósito al cual se transfiere la ruta.

$SwapRutaD$: Este operador realiza un intercambio de dos rutas pertenecientes a diferentes depósitos. El criterio para hacer movimientos de este tipo es que la solución obtenida sea factible, es decir, que no se supera la capacidad de los depósitos involucrados en el intercambio.

$SwapSubPro$: Finalmente se propone un nuevo operador con el fin de cambiar la configuración de rutas propias y rutas subcontradas de los depósitos. Este operador hace un intercambio aleatorio de una ruta abierta con una ruta cerrada. En términos del modelo MDVRPPC se traslada un vehículo propio de un depósito a otro reemplazando una ruta subcontratada y los clientes que inicialmente eran atendidos por este pasan a ser atendidos por un vehículo subcontratado.

19 Análisis de Resultados

En esta sección se propone la aplicación de la metodología planteada en las secciones 15, 16, 17 y 18 a un conjunto de instancias planteadas por (Toro-Ocampo et al., 2016). En su mayoría son instancias del modelo MDVRP modificadas y adaptadas para el modelo de flota propia y subcontratada propuestas inicialmente por (Jean-Francois Cordeau, Gendreau, & Laporte, 1997). Para realizar la comparación de resultados se utilizarán las diferencias entre la solución de la metodología propuesta y el modelo exacto. La información que se utilizó para realizar la comparación con el modelo exacto es la solución óptima (o la mejor solución encontrada), el tiempo de ejecución y el GAP de la solución. Es importante mencionar que para el modelo exacto, el GAP corresponde a la diferencia entre el límite superior y el límite inferior en el método *Branch and Cut.* Lo anterior implica que, en los casos donde se obtuvo un GAP superior al 0%, el modelo exacto no puede garantizar la solución óptima.

La complejidad y el gasto computacional para resolver cualquier instancia aumenta con el aumento del número de nodos que esta tenga. Esto hace necesario disponer de un equipo de cómputo que ejecute todas las instrucciones de los algoritmos planteados de manera eficiente. Todo el código necesario para implementar las metodologías aquí propuestas se desarrolló en el software Matlab® y ya que este está especialmente orientado al manejo de matrices y vectores, todas la funciones y scripts se hicieron considerando esta condición.

El algoritmo fue ejecutado en un equipo con un procesador AMD Ryzen 5 1400 con frecuencia base de 3.2 Ghz y Turbo de hasta 3.4 Ghz y con sistema operativo Windows 10 de 64b bits.

19.1 Codificación

La codificación de los problemas de ruteo es especialmente importante ya que esta puede variar según la variante del VRP que se esté tratando. Las variables más importantes al momento de considerar la codificación tienen que ver con la configuración de las rutas. A partir de esta información se debe poder identificar los clientes asignados a cada una, el tipo de ruta, el total de la demanda, el depósito al cual pertenecen y su costo. Es importante considerar la naturaleza dinámica de las rutas ya que las metodologías que resuelven los problemas de ruteo constantemente realizan intercambios de clientes entre las rutas. Lo anterior hace necesario dimensionar las matrices y vectores desde un principio de manera que soporten la naturaleza variable la configuración de la solución de este tipo de problemas.

En el problema específico de múltiples depósitos se tienen nodos adicionales correspondientes a los depósitos, por lo que se define una matriz con la información o la unión del conjunto de clientes I con el conjunto de depósitos J. Así, si se tiene un conjunto de n clientes y m depósitos la matriz con la información de todos los nodos tendrá la siguiente dimensión.

$$(n+m, 3)$$

Las tres columnas de la anterior matriz contienen la información de la demanda (y capacidad en el caso de los depósitos) y las coordenadas x y y de cada uno de los nodos. Los clientes entonces siempre ocupan las primeras n filas de la matriz y los depósitos están asignados en el rango $[n+1, n+m]$.

Adicionalmente se definen dos matrices adicionales que almacenan respectivamente la información de la configuración de clientes de cada ruta y las características adicionales de las rutas.

Matriz SoloRutas: Es una matriz que únicamente contiene la información de los clientes asignados a cada una de las rutas. Así, si se tienen k rutas, las dimensiones de esta matriz son (k, n). Esta manera se define desde un principio la máxima longitud de una ruta en el caso de que contuviera a todos los clientes. Todas las demás posiciones de la matriz donde no se tienen nodos asignados tienen el valor de cero.

Matriz RutasAhorro: Esta matriz tiene el restante de la información asociada a cada ruta. Así, sus dimensiones son $(k, 6)$. En las 6 columnas de esta matriz se almacena el depósito al que pertenece la ruta, costo, demanda acumulada, el tipo de ruta, coordenadas x y y del centroide.

De esta manera las dimensiones de las matrices que guardan la información de la solución se mantienen constantes simplificando los cálculos y evitando el impacto computacional.

19.2 Parámetros para la implementación del algoritmo

En esta sección se presenta los valores de los parámetros configurables del modelo. De la correcta definición de estos parámetros depende en gran medida la efectividad del algoritmo para llegar a soluciones de calidad.

$N_p = 0.8 * \sum_{i \in I} qi \,/Q$	(19.1)	Número de vehículos propios.
$F_{Sub} = 2.0$	(19.2)	Factor de sobre costo por utilizar la flota subcontratada.
$DistIntercambio = w * R$	(19.3)	Distancia permitida para intercambios de clientes entre rutas (Véase Figura 14)
$MaxIter = 70 * V$	(19.4)	Máximo de iteraciones globales.
$MaxIterILS = \#Nodos + 20 * V$	(19.5)	Máximo de iteraciones del ILS.

Tabla 30 Valores de Parámetros Configurables

Donde:

$\sum_{i \in I} qi$ = Sumatoria de las demandas de todos los clientes.

Q =Capacidad de los Vehículos

F_{Sub}= Se parte del supuesto que la utilización de la flota subcontratada es equivalente al costo normal por un valor de penalización.

R =Distancia entre el centroide de la ruta y el depósito al que pertenece.

w =Número de veces la longitud R donde se define el radio de los posibles intercambios. Para este caso se definió su valor en 2.

V =Número de vehículos en la solución, tanto vehículos propios como subcontratados.

19.3 Resultados para Instancias seleccionadas

La cantidad de vehículos propios Np se define utilizando la ecuación (19.1), donde para las instancias escogidas se define que se dispone de los vehículos suficientes para satisfacer al 80% de la demanda de los clientes. De esta manera se deben asignar vehículos extra para atender el 20% restante de la demanda. En este caso los vehículos tienen capacidad homogénea.

Una de las principales etapas para la solución de los problemas con múltiples depósitos es asignación de vehículos a los depósitos previo a la construcción de la solución inicial. Como se explicó en la sección 15 se adoptaron 7 métodos de "clusterización" para la asignación de clientes. Cada uno de ellos por medio de un criterio de distancia, crean grupos que posteriormente se asignan a los depósitos más cercanos. Así, se suelen obtener diferentes configuraciones de rutas dependiendo del método aglomerativo escogido. Por lo anterior, se estudiará los resultados de cada método por separado ya que la calidad de la solución de algunas instancias está relacionada con el proceso de "clusterización" elegido.

Los resultados mostrados a continuación muestran los resultados de la solución inicial obtenida del algoritmo de ahorros modificado aplicado a cada método de "clusterización". Luego la solución promedio y la mejor solución a partir de 10 corridas aplicando la búsqueda local iterada o ILS. Finalmente, se muestra las comparativas con el modelo exacto por medio del GAP usando las ecuaciones (19.6), (19.7) y (19.8).

$$GAP_1 = \frac{Solución\ Incial - Solución\ Modelo\ Exacto}{Solución\ Modelo\ Exacto} \quad (19.6)$$

$$GAP_2 = \frac{Mejor\ Solución - Solución\ Modelo\ Exacto}{Solución\ Modelo\ Exacto} \quad (19.7)$$

$$GAP_3 = \frac{Promedio\ Soluciones - Solución\ Modelo\ Exacto}{Solución\ Modelo\ Exacto} \quad (19.8)$$

Instancia: P01-20-4 Solución Modelo Exacto: 300 (GAP 0%) Tiempo: 212 (s)

Método de "clusterización"	Solución Inicial	GAP_1	Promedio Soluciones (10 Corridas)	Tiempo Promedio (s) (10 Corridas)	Mejor Solución	GAP_2	GAP_3
SingleLinkage	314	4.67%	300.1	24.1	300	0.00%	0.03%
CompleteLinkage	315	5.00%	300	21.6	300	0.00%	0.00%
AverageLinkage No Ponderado	326	8.67%	300	26.3	300	0.00%	0.00%
AverageLinkage Ponderado	314	4.67%	300	23.2	300	0.00%	0.00%
Centroide Ponderado	314	4.67%	300	24.1	300	0.00%	0.00%
Centroide No Ponderado	315	5.00%	300	20.9	300	0.00%	0.00%
Ward	315	5.00%	300	21.8	300	0.00%	0.00%

Tabla 31 Resultados Instancia P01-20-4

Como se muestra en la Tabla 31, se puede observar que el método de "clusterización" no tiene impacto significativo sobre el algoritmo para alcanzar la solución óptima. Cabe destacar que un método como el "AverageLinkage No Ponderado" no necesariamente es inconveniente pese a que tiene un $GAP_1 = 8.67\%$. La efectividad de un método de "clusterización" se mide mejor con el GAP_3 que mide en promedio las soluciones alcanzadas después de aplicar el ILS. En general se puede concluir que esta metodología alcanzó soluciones de calidad ya que los resultados del GAP_3 en casi todos los casos es del 0%.

Instancia: P01-25-4		Solución Modelo Exacto: 368 (GAP 0%)		Tiempo: 27047 (s)			
Método de "clusterización"	Solución Inicial	GAP_1	Promedio Soluciones (10 Corridas)	Tiempo Promedio (s) (10 Corridas)	Mejor Solución	GAP_2	GAP_3
SingleLinkage	422	14.67%	368.5	45.5	368	0.00%	0.14%
CompleteLinkage	426	15.76%	368	38.3	368	0.00%	0.00%
AverageLinkage No Ponderado	417	13.32%	368.4	44.0	368	0.00%	0.11%
AverageLinkage Ponderado	418	13.59%	368.3	42.9	368	0.00%	0.08%
Centroide Ponderado	426	15.76%	368	43.3	368	0.00%	0.00%
Centroide No Ponderado	426	15.76%	368	41.8	368	0.00%	0.00%
Ward	426	15.76%	368	39.7	368	0.00%	0.00%

Tabla 32 Resultados Instancia P01-25-4

En el caso de la segunda instancia analizada las soluciones iniciales superan al 13% en todos los métodos utilizados (Véase Tabla 32). No obstante, todas llegan a la solución óptima en casi todas las corridas. Por otro lado, el GAP_3 se mantiene cercano al 0% casi siempre en todos los métodos.

Instancia: P01-30-4		Solución Modelo Exacto: 411 (GAP 2%)		Tiempo: 42894 (s)			
Método de "clusterización"	Solución Inicial	GAP_1	Promedio Soluciones (10 Corridas)	Tiempo Promedio (s) (10 Corridas)	Mejor Solución	GAP_2	GAP_3
SingleLinkage	453	10.22%	412.8	57.4	412	0.24%	0.44%
CompleteLinkage	453	10.22%	413.3	56.8	413	0.49%	0.56%
AverageLinkage No Ponderado	453	10.22%	413.1	56.6	413	0.49%	0.51%
AverageLinkage Ponderado	453	10.22%	413.1	56.8	413	0.49%	0.51%
Centroide Ponderado	453	10.22%	413	52.2	413	0.49%	0.49%
Centroide No Ponderado	453	10.22%	413.1	53.9	413	0.49%	0.51%
Ward	453	10.22%	413	55.7	412	0.24%	0.49%

Tabla 33 Resultados Instancia P01-30-4

En contraste a las anteriores, en la tercera instancia analizada no se alcanza el óptimo en ninguno de los 7 métodos de "clusterización". No obstante, como se puede observar en la Tabla 33 las mejores soluciones no tienen un GAP_2 superior al 0.5% y el GAP_3 se mantiene por debajo del 1%.

Instancia: P01-30-3			Solución Modelo Exacto: 438 (GAP 2%)	Tiempo: 61809 (s)			
Método de "clusterización"	Solución Inicial	GAP_1	Promedio Soluciones (10 Corridas)	Tiempo Promedio (s) (10 Corridas)	Mejor Solución	GAP_2	GAP_3
SingleLinkage	470	7.31%	438	54.0	438	0.00%	0.00%
CompleteLinkage	467	6.62%	438	59.1	438	0.00%	0.00%
AverageLinkage No Ponderado	479	9.36%	438	55.7	438	0.00%	0.00%
AverageLinkage Ponderado	479	9.36%	438	58.4	438	0.00%	0.00%
Centroide Ponderado	479	9.36%	438	58.4	438	0.00%	0.00%
Centroide No Ponderado	479	9.36%	438	56.9	438	0.00%	0.00%
Ward	462	5.48%	438	55.2	438	0.00%	0.00%

Tabla 34 Resultados Instancia P01-30-3

En la Tabla 34 se muestran los resultados para la cuarta instancia considerada. De manera similar a las dos primeras, se obtienen soluciones de calidad de manera que el GAP_3 se mantiene para todos los métodos de "clusterización" en 0%. Esto quiere decir que se alcanzó la solución óptima en todas las corridas realizadas.

Finalmente, se presentan los resultados para la quinta instancia. En este caso en particular el modelo exacto arrojó una solución de 521 con un GAP del 6%. Lo anterior quiere decir que la búsqueda de la solución se detuvo sin alcanzar el óptimo. Por esta razón la metodología propuesta en este trabajo no solo alcanzo la solución del modelo exacto, sino que obtuvo soluciones hasta un 4.03% por debajo. Como se puede observar en la Tabla 35, todos los $GAP_3 s$ son negativos lo que quiere decir que casi siempre se mejoraba la solución obtenida por el modelo exacto.

Instancia: P02-50-4			Solución Modelo Exacto: 521 (GAP 6%)	Tiempo: 50000 (s)			
Método de "clusterización"	Solución Inicial	GAP_1	Promedio Soluciones (10 Corridas)	Tiempo Promedio (s) (10 Corridas)	Mejor Solución	GAP_2	GAP_3
SingleLinkage	583	11.90%	500.4	575.7	500	-4.03%	-3.95%
CompleteLinkage	583	11.90%	500	567.5	500	-4.03%	-4.03%
AverageLinkage No Ponderado	583	11.90%	506.5	617.1	500	-4.03%	-2.78%
AverageLinkage Ponderado	570	9.40%	500.2	394.8	500	-4.03%	-3.99%
Centroide Ponderado	561	7.68%	505.5	610.5	500	-4.03%	-2.98%
Centroide No Ponderado	583	11.90%	500.4	567.2	500	-4.03%	-3.95%
Ward	583	11.90%	503.9	588.6	500	-4.03%	-3.28%

Tabla 35 Resultados Instancia P02-50-4

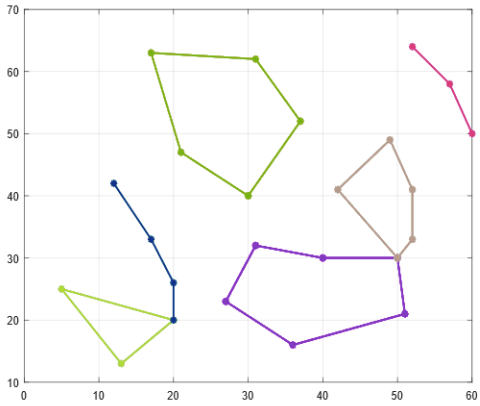
En general se puede decir que, pese a que los métodos de "clusterización" generan soluciones iniciales diferentes, el diseño del algoritmo permite escapar de óptimos locales y llegar a soluciones óptimas o muy cercanas al óptimo.

A continuación, se muestra la configuración de clientes en cada una de las rutas propias y subcontratadas correspondientes a la mejor solución encontrada (Véase la Tabla 36).

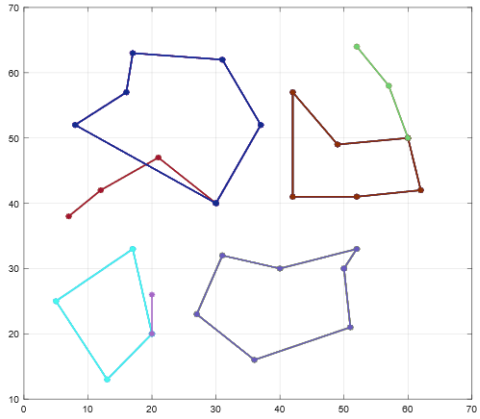
Instancia	Np	Rutas Propias	Rutas Subcontratadas
P01-20-4	4	21 19 13 21 22 6 7 8 1 22 23 5 12 17 15 10 23 23 9 16 2 11 23	21 4 18 14 24 20 3
P01-25-4	4	26 19 13 18 26 27 1 8 7 23 24 27 28 10 15 17 12 5 9 28 29 21 16 11 22 2 29	26 4 27 6 14 25 29 20 3
P01-30-4	5	32 6 24 25 14 32 32 23 7 26 8 27 32 33 5 17 15 10 30 9 33 34 20 3 28 22 1 29 34 34 21 16 11 2 34	31 19 31 4 18 13 32 12 0
P01-30-3	5	31 17 18 13 19 31 32 1 22 28 3 20 32 32 24 23 7 26 8 32 33 9 30 10 15 5 33 33 16 21 29 2 11 33	31 4 32 27 6 14 25 32 12
P02-50-4	4	51 17 37 15 33 45 44 42 19 40 41 13 4 51 52 27 6 48 23 24 43 7 26 8 1 32 46 52 53 9 50 34 30 39 10 49 5 11 38 53 54 20 35 36 3 28 31 22 2 16 21 29 54	52 12 47 18 14 25

Tabla 36 Resultados de Rutas de las Instancias Consideradas

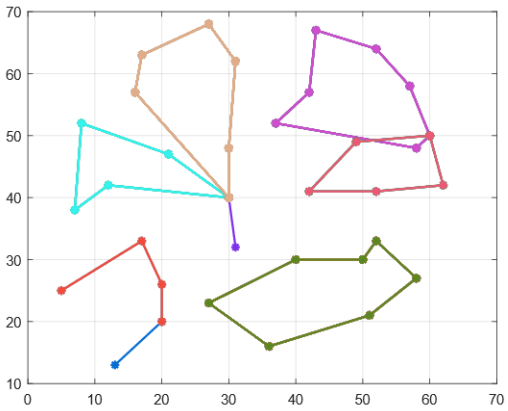
Las gráficas comprendidas entre Gráfica 1 y Gráfica 5 muestran las mejores soluciones alcanzadas por la metodología propuesta.



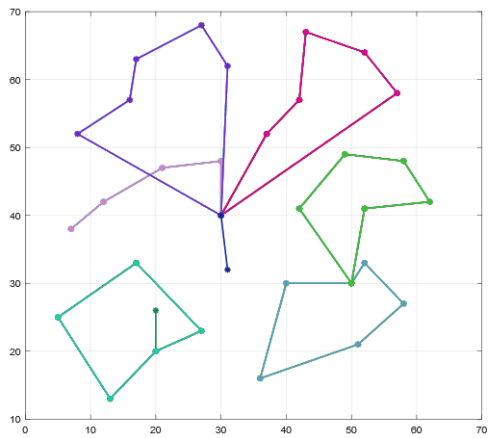
Gráfica 1 Mejor Solución Encontrada P01-20-4



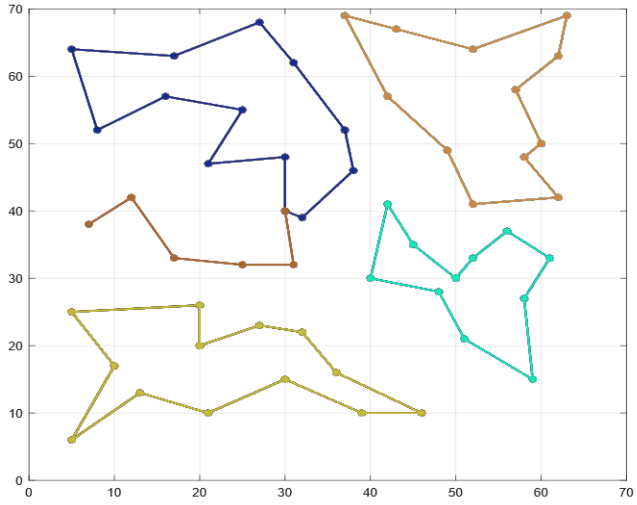
Gráfica 2 Mejor Solución Encontrada P01-25-4



Gráfica 3 Mejor Solución Encontrada P01-30-4



Gráfica 4 Mejor Solución Encontrada P01-30-3



Gráfica 5 Mejor Solución Encontrada P02-50-4

19.4 Resultados para instancias de mayor complejidad

El tiempo de ejecución del algoritmo está directamente relacionado con la cantidad de nodos que tenga la instancia bajo estudio. En esta sección se presentan los resultados para instancias de mayor tamaño (más de 50 clientes) propuestas en (Jean-Francois Cordeau et al., 1997). Bajo el mismo criterio utilizado en la sección 19.3, se estimó la cantidad de vehículos propios disponibles como la cantidad necesaria para atender el 80% de la demanda. El restante de la demanda debe ser atendida por vehículos subcontratados. El número de pruebas se estableció en 5 y se limitó la cantidad máxima de iteraciones a 50 debido al tiempo requerido. Los resultados para cada uno de los métodos de "clusterización" utilizados se muestran desde la Tabla 37 a la Tabla 52.

Instancia: P03-75-4

Método de "clusterización"	Solución Inicial	Promedio Soluciones (5 Corridas)	Tiempo Promedio (s) (5 Corridas)	Mejor Solución
SingleLinkage	797	678.4	1033.35	674
CompleteLinkage	767	670.6	1234.73	668
AverageLinkage No Ponderado	785	679	1272.73	673
AverageLinkage Ponderado	766	671.2	952.77	668
Centroide Ponderado	766	673.6	1176.59	672
Centroide No Ponderado	764	678.4	1265.24	676
Ward	767	678.8	1106.83	676

Tabla 37 Resultados instancia P03-75-5

Instancia: P03-75-4 Np =8	
Rutas Propias	76 75 4 68 6 51 17 26 76 77 29 45 30 2 74 21 47 77 77 36 69 71 60 70 20 37 15 57 5 77 78 35 19 54 13 27 52 8 7 78 79 72 58 10 31 25 55 18 50 3 79 79 44 32 9 39 12 40 79 80 33 63 16 49 24 23 56 41 80 80 73 62 28 61 22 64 42 43 80
Rutas Subcontratadas	76 67 34 46 77 48 78 14 59 78 53 11 66 65 38 80 1

Tabla 38 Resultados de Rutas de la Instancia P03-75-5

Instancia: P04-100-2

Método de "clusterización"	Solución Inicial	Promedio Soluciones (5 Corridas)	Tiempo Promedio (s) (5 Corridas)	Mejor Solución
SingleLinkage	1141	1025.8	2276.27	1023
CompleteLinkage	1167	1029.6	2208.67	1028
AverageLinkage No Ponderado	1141	1023.6	2291.79	1017
AverageLinkage Ponderado	1143	1026.8	2213.24	1020
Centroide Ponderado	1141	1028.8	2186.45	1022
Centroide No Ponderado	1143	1027	2248.21	1022
Ward	1176	1026.6	2144.26	1024

Tabla 39 Resultados instancia P04-100-2

Instancia: P04-100-2 Np =12

Rutas Propias	101 74 23 67 39 56 101
	101 2 41 22 75 72 73 101
	101 84 17 86 16 61 93 92 101
	101 12 80 29 24 54 55 25 4 21 101
	101 57 15 43 14 38 44 91 100 42 101
	101 58 53 27 28 26 40 101
	102 52 18 83 60 5 99 96 6 89 101
	102 70 30 32 90 63 62 10 101
	102 51 9 81 33 79 3 101
	102 82 8 45 46 48 7 88 101
	102 20 66 71 65 35 34 78 69 101
	102 19 47 36 49 64 11 101
Rutas Subcontratadas	102 19 47 36 49 64 11 101
	101 13 94 95 59
	101 87 97 37 98 85
	102 31
	102 1 50 76 77 68

Tabla 40 Resultados de Rutas de la Instancia P04-100-2

Instancia: P05-100-2

Método de "clusterización"	Solución Inicial	Promedio Soluciones (5 Corridas)	Tiempo Promedio (s) (5 Corridas)	Mejor Solución
SingleLinkage	919	786.8	8773.90	784
CompleteLinkage	924	776.8	8766.50	772
AverageLinkage No Ponderado	917	769.8	9557.20	767
AverageLinkage Ponderado	880	778.4	7892.98	769
Centroide Ponderado	940	773	9023.14	765
Centroide No Ponderado	924	769.8	8385.16	766
Ward	890	775.4	8707.08	770

Tabla 41 Resultados instancia P05-100-2

Instancia: P05-100-2 Np =6	
Rutas Propias	101 6 94 92 37 98 100 91 44 14 38 86 16 61 17 84 101 101 18 52 7 88 31 10 62 11 19 48 82 8 83 101 101 60 89 27 69 1 70 30 32 90 63 64 49 36 47 46 45 101 102 54 4 74 22 41 15 43 42 57 2 13 58 53 28 12 102 102 24 29 78 34 35 65 71 66 20 51 9 81 33 50 76 102 102 55 25 39 67 23 56 75 72 73 21 40 26 102
Rutas Subcontratadas	101 5 85 93 59 99 96 95 97 87 102 80 68 77 3 79

Tabla 42 Resultados de Rutas de la Instancia P05-100-2

Instancia: P06-100-3

Método de "clusterización"	Solución Inicial	Promedio Soluciones (5 Corridas)	Tiempo Promedio (s) (5 Corridas)	Mejor Solución
SingleLinkage	1012	901.6	1588.48	900
CompleteLinkage	986	901.8	1798.29	901
AverageLinkage No Ponderado	991	902.4	1714.23	899
AverageLinkage Ponderado	993	901.4	1835.49	896
Centroide Ponderado	1008	901.2	1778.58	896
Centroide No Ponderado	999	903.6	1867.86	898
Ward	1005	902.6	1750.78	898

Tabla 43 Resultados instancia P06-100-3

Instancia: P06-100-3 Np =12	
Rutas Propias	101 14 42 43 15 57 2 87 97 37 101
	101 91 44 38 86 16 101
	101 61 84 17 45 46 8 18 83 60 5 101
	102 56 23 67 39 102
	102 40 58 53 28 12 26 102
	102 54 80 68 29 24 55 25 4 102
	102 72 75 22 41 74 73 21 102
	103 88 7 48 82 52 27 69 103
	103 70 71 65 66 20 30 103
	103 33 81 78 34 35 9 51 103
	103 19 47 36 49 64 11 103
	103 10 32 90 63 62 31 103
Rutas Subcontratadas	101 93 59 99 96 6 89
	101 100 98 92 95 94 13
	101 85
	103 1 50 76 77 3 79

Tabla 44 Resultados de Rutas de la Instancia P06-100-3

Instancia: P07-100-4

Método de "clusterización"	Solución Inicial	Promedio Soluciones (5 Corridas)	Tiempo Promedio (s) (5 Corridas)	Mejor Solución
SingleLinkage	1041	899.6	1737.80	891
CompleteLinkage	1037	914	1585.43	910
AverageLinkage No Ponderado	1041	904.2	1689.56	894
AverageLinkage Ponderado	1041	891.8	1329.69	890
Centroide Ponderado	1059	904.6	1744.43	892
Centroide No Ponderado	1034	907.2	1369.60	897
Ward	1054	907.4	1528.05	898

Tabla 45 Resultados instancia P07-100-4

Instancia: P07-100-4 Np =12

Rutas Propias	101 84 61 16 86 17 45 101
	101 18 7 19 11 64 49 36 46 8 101
	102 54 4 21 40 26 12 102
	102 24 29 78 34 35 81 33 50 76 102
	102 55 25 39 67 23 56 102
	103 58 53 89 6 94 13 103
	103 73 72 74 75 22 41 2 103
	103 57 15 43 14 38 44 91 100 42 103
	103 95 96 59 92 97 87 103
	104 31 88 52 27 28 1 69 104
	104 10 62 63 90 32 30 70 104
	104 51 9 71 65 66 20 104
Rutas Subcontratadas	101 60 99 93 85 98 37
	101 83 82 48 47
	101 5
	102 80 68 77 3 79

Tabla 46 Resultados de Rutas de la Instancia P07-100-4

Instancia: P12-80-2

Método de "clusterización"	Solución Inicial	Promedio Soluciones (5 Corridas)	Tiempo Promedio (s) (5 Corridas)	Mejor Solución
SingleLinkage	1471	1399.2	3106.01	1390
CompleteLinkage	1473	1389.4	3209.05	1379
AverageLinkage No Ponderado	1521	1408.8	3583.70	1392
AverageLinkage Ponderado	1510	1392	3429.90	1390
Centroide Ponderado	1474	1398	2946.55	1398
Centroide No Ponderado	1452	1394.4	3499.60	1390
Ward	1571	1391.6	3397.85	1390

Tabla 47 Resultados instancia P12-80-2

Instancia: P12-80-2 Np=6	
Rutas Propias	81 3 11 19 27 35 37 29 21 13 5 81 81 4 12 20 28 36 38 73 30 22 14 6 81 81 1 9 17 25 33 34 26 18 10 2 81 82 55 63 71 79 78 70 62 54 52 60 68 76 65 57 49 82 82 50 58 66 74 39 31 23 15 16 24 32 40 75 67 59 51 82 82 45 53 61 69 77 80 72 64 56 48 82
Rutas Subcontratadas	81 7 8 82 47 46 82 44 41 42 43

Tabla 48 Resultados de Rutas de la Instancia P12-80-2

Instancia: P15-160-4

Método de "clusterización"	Solución Inicial	Promedio Soluciones (5 Corridas)	Tiempo Promedio (s) (5 Corridas)	Mejor Solución
SingleLinkage	3015	2666	21071.15	2657
CompleteLinkage	2893	2652	19115.90	2630
AverageLinkage No Ponderado	2920	2650.2	19562.23	2641
AverageLinkage Ponderado	3103	2659	19084.51	2653
Centroide Ponderado	3148	2699.6	17397.05	2671
Centroide No Ponderado	2895	2642.2	19487.75	2631
Ward	3130	2634.6	17733.23	2627

Tabla 49 Resultados instancia P15-160-4

Instancia: P15-160-4 Np =12

Rutas Propias	161 6 12 20 28 36 33 25 17 9 34 26 18 10 161
	161 14 22 30 38 73 65 57 49 50 58 66 74 39 31 23 15 161
	161 4 1 2 3 5 161
	162 51 59 67 75 40 32 150 158 113 105 97 89 100 108 116 77 69 61 53 162
	162 44 52 60 68 76 78 70 62 54 46 162
	162 48 56 64 72 79 71 63 55 47 162
	163 88 96 104 112 120 117 109 101 93 85 163
	163 86 94 102 110 80 118 119 111 103 95 87 163
	163 83 91 99 107 115 160 152 144 136 151 159 114 106 98 90 163
	164 129 137 145 153 35 27 19 11 13 21 29 37 156 148 140 132 164
	164 127 135 143 142 134 126 124 164
	164 133 141 149 157 155 147 139 131 154 146 138 130 121 164
Rutas Subcontratadas	161 7 8 16 24
	162 45 43 42 41
	163 82 81 84 92
	164 122 123 125 128

Tabla 50 Resultados de Rutas de la Instancia P15-160-4

Instancia: P18-240-6

Método de “clusterización”	Solución Inicial	Promedio Soluciones (2 Corridas)	Tiempo Promedio (s) (2 Corridas)	Mejor Solución
SingleLinkage	4581	3926	59518.22	3917
CompleteLinkage	4344	3963.5	53211.04	3953
AverageLinkage No Ponderado	4309	3986.5	45938.60	3980
AverageLinkage Ponderado	4338	3950.5	46212.99	3941
Centroide Ponderado	4891	4006.5	41779.30	3996
Centroide No Ponderado	4616	3964	46384.07	3940
Ward	5110	3978.5	42831.71	3956

Tabla 51Resultados instancia P18-240-6

Instancia: P18-240-6 Np =19	
Rutas Propias	241 4 1 9 18 10 2 241 241 14 22 30 38 73 65 57 49 50 58 66 74 39 31 23 15 241 241 16 24 32 40 75 67 105 113 158 150 142 134 140 148 156 37 29 21 13 241 241 11 19 27 35 153 198 240 232 224 223 231 239 34 26 230 238 33 25 17 36 28 20 12 241 241 6 7 8 5 3 241 242 55 63 71 79 78 70 62 54 76 68 60 52 41 242 242 44 46 47 48 45 242 242 53 61 69 77 116 108 100 92 94 102 110 118 80 72 64 56 242 243 88 96 104 112 120 119 111 103 95 87 243 243 93 101 109 117 91 99 107 115 160 152 144 143 151 159 114 106 98 90 243 244 128 136 157 149 141 133 123 125 244 244 131 139 147 155 200 192 184 176 175 183 191 199 154 146 138 130 244 245 173 181 189 197 195 187 179 171 194 186 178 170 161 245 245 164 172 180 190 182 174 166 245 245 162 163 165 168 167 245 246 206 214 222 236 228 220 212 204 246 246 202 210 218 226 234 233 225 217 209 201 246 246 203 211 219 227 235 193 185 177 169 188 196 237 229 221 213 246 246 205 208 216 215 207 246
Rutas Subcontratadas	242 42 43 51 59 243 85 83 82 81 89 97 243 84 86 244 122 121 129 137 145 244 124 132 126 127 135

Tabla 52 Resultados de Rutas de la Instancia P18-240-6

20 Conclusiones

Un modelo basado en técnicas metaheurísticas fue propuesto para la solución del problema de múltiples depósitos con flota propia y subcontratada. Este modelo tuvo como base la aplicación de la búsqueda local iterada o ILS que para el problema MDVRPPC e implicó la aplicación de operadores inter-depósito, operadores inter-ruta y operadores intra-ruta. En cuanto a la generación de las soluciones para inicializar el ILS, se utilizaron técnicas de "clusterización" y su posterior procesamiento con una modificación del algoritmo de ahorros clásico. En comparación con los resultados presentes en la literatura, en la mayoría de los casos el algoritmo propuesto demostró ser lo suficientemente potente para escapar de óptimos locales y llegó a soluciones óptimas casi independientemente del método de "clusterización" usado.

Después de una revisión de la literatura asociada al modelo con flota propia y subcontratada, se encontró que muy pocas publicaciones se han hecho para tratar el problema con múltiples de depósitos. En su mayoría se tienen publicaciones de un único depósito o de ruteo únicamente abierto para el caso multidepósito. Las publicaciones más importantes que abordan el tema son la de (Toro-Ocampo et al., 2016) y la presentada en (Azadeh & Farrokhi-Asl, 2017) cuyos modelos matemáticos fueron analizados en las sección 10.7. No obstante, se han publicado metodologías muy útiles que tienen su aplicación en el MDVRPPC. De acuerdo con lo encontrado, es posible aplicar algunos métodos de "clusterización" y una modificación del algoritmo de ahorros para la construcción de una solución inicial. Por otro lado, se destacó la implementación de operadores para el intercambio de clientes entre rutas y al interior de las rutas para el mejoramiento de la solución.

Se cumple con la propuesta de implementar una metodología metaheurística como alternativa de solución a los modelos exactos. Este enfoque provee una serie de ventajas sobre los modelos tipo exacto al alcanzar soluciones óptimas o muy cercanas al óptimo en tiempos computacionalmente aceptables. Por otro lado, se muestra la utilidad de procedimientos, como la búsqueda local iterada, para la solución de este tipo de problemas de naturaleza combinatorial. Esta metodología resulta ser sustancialmente eficiente una vez se hayan definido las estructuras correctas y se configuren adecuadamente los parámetros del modelo. Esto hace del ILS una metodología que logra un equilibro entre velocidad y calidad de las soluciones.

Los resultados obtenidos por la metodología propuesta se consideran satisfactorios logrando en promedio GAPs menores al 1% en todas las instancias analizadas. Así mismo, en casi todas las instancias se alcanza al menos una vez el óptimo global del problema. En otras instancias como la última en consideración (en la que el modelo exacto no encuentra la solución global) no solo se iguala la solución sino que se mejora, logrando GAPs negativos de acuerdo a las ecuaciones (19.7) y (19.8).

Una característica que es importante destacar, es que la metaheurística propuesta puede ser fácilmente adaptada para la solución de algunas variantes del VRP. Así, únicamente configurando la cantidad de vehículos se pueden resolver problemas como el MDVRP, MDOVRP y el MDVRPPC. Por otro lado, cuando se define la cantidad de depósitos igual a 1 y se modifica la cantidad de vehículos disponible, se pueden resolver problemas como el VRP clásico, OVRP y VRPPC. Por lo anterior, se propone para estudios futuros implementar las adecuaciones suficientes permitan resolver los modelos mencionados y contrastar sus resultados con los presentes en la literatura, ya que se espera que su eficiencia sea la misma que con la variante MDVRPPC tratada en este trabajo.

En otros trabajos futuros y con miras a incorporar características más ajustadas a la realidad, se sugiere el desarrollo de modelos donde uno o más componentes sean estocásticos. Estos modelos pueden implicar que las demandas de los clientes o los tiempos de recorrido entre clientes sean variables aleatorias. Este tipo de modelos con componentes probabilísticos se consideran interesantes debido a su complejidad y a que en la actualidad no se tienen muchas publicaciones sobre el tema.

21 Bibliografía

Afshar-Nadjafi, B., & Afshar-Nadjafi, A. (2014). A constructive heuristic for time-dependent multi-depot vehicle routing problem with time-windows and heterogeneous fleet. *Journal of King Saud University - Engineering Sciences*, *29*(1). https://doi.org/10.1016/j.jksues.2014.04.007

Azadeh, A., & Farrokhi-Asl, H. (2017). The close–open mixed multi depot vehicle routing problem considering internal and external fleet of vehicles. *Transportation Letters*, *7867*(January), 1–15. https://doi.org/10.1080/19427867.2016.1274468

Baldacci, R., Bartolini, E., Mingozzi, A., & Valletta, A. (2011). An Exact Algorithm for the Period Routing Problem. *Operations Research*, *59*(1), 228–241. https://doi.org/10.1287/opre.1100.0875

Baldacci, R., & Mingozzi, A. (2009). A unified exact method for solving different classes of vehicle routing problems. *Mathematical Programming*, *120*(2), 347–380. https://doi.org/10.1007/s10107-008-0218-9

Ball, M. O., Golden, B. L., Assad, A. A., & Bodin, L. D. (1983). PLANNING FOR TRUCK FLEET SIZE IN THE PRESENCE OF A COMMON-CARRIER OPTION. *Decision Sciences*, *14*(1), 103–120. https://doi.org/10.1111/j.1540-5915.1983.tb00172.x

Barreto, S., Ferreira, C., Paixão, J., & Santos, B. S. (2007). Using clustering analysis in a capacitated location-routing problem. *European Journal of Operational Research*, *179*(3), 968–977. https://doi.org/10.1016/j.ejor.2005.06.074

Bell, W. J., Dalberto, L. M., Fisher, M. L., Greenfield, A. J., Jaikumar, R., Kedia, P., … Prutzman, P. J. (1983). Improving the Distribution of Industrial Gases with an On-Line Computerized Routing and Scheduling Optimizer. *Interfaces*, *13*(6), 4–23. https://doi.org/10.1287/inte.13.6.4

Bolduc, M-C, Renaud, J., Boctor, F., & Laporte, G. (2008). A perturbation metaheuristic for the vehicle routing problem with private fleet and common carriers. *Journal of the Operational Research Society*, *59*(6), 776–787. https://doi.org/10.1057/palgrave.jors.2602390

Bolduc, Marie-Claude, Renaud, J., & Boctor, F. (2007). A heuristic for the routing and carrier selection problem. *European Journal of Operational Research*, *183*(2), 926–932. https://doi.org/10.1016/j.ejor.2006.10.013

Braekers, K., Caris, A., & Janssens, G. K. (2014). Exact and meta-heuristic approach for a general heterogeneous dial-a-ride problem with multiple depots. *Transportation Research Part B: Methodological*, *67*, 166–186. https://doi.org/10.1016/j.trb.2014.05.007

Braekers, K., Ramaekers, K., & Van Nieuwenhuyse, I. (2016). The vehicle routing problem: State of the art classification and review. *Computers & Industrial Engineering*, *99*, 300–313. https://doi.org/10.1016/j.cie.2015.12.007

Cassidy, P and Bennett, H. (1972). Multi-Depot Vehicle Scheduling System. *Operational Research Quarterly*, *23*, 151–163.

Chao, M., Golden, B., & Wasil, E. (1993). A New Heuristic for the Multi-Depot Vehicle Routing Problem that Improves upon Best-Known Solutionso Title. *American Journal of Mathematical and Management Sciences*, *13*(3–4), 371–406.

Chu, C.-W. (2005). A heuristic algorithm for the truckload and less-than-truckload problem. *European Journal of Operational Research*, *165*(3), 657–667. https://doi.org/10.1016/j.ejor.2003.08.067

Clarke, G., & Wright, J. W. (1964). Scheduling of Vehicles from a Central Depot to a Number of Delivery Points. *Operations Research*, *12*(4), 568–581. https://doi.org/10.1287/opre.12.4.568

Contardo, C., & Martinelli, R. (2014). A new exact algorithm for the multi-depot vehicle routing problem under capacity and route length constraints. *Discrete Optimization*, *12*, 129–146. https://doi.org/10.1016/j.disopt.2014.03.001

Cordeau, Jean-Francois, Gendreau, M., & Laporte, G. (1997). A tabu search heuristic for periodic and multi-depot vehicle routing problems. *Networks*, *30*(2), 105–119. https://doi.org/10.1002/(SICI)1097-0037(199709)30:2<105::AID-NET5>3.0.CO;2-G

Cordeau, Jean-François, & Maischberger, M. (2012). A parallel iterated tabu search heuristic for vehicle routing problems. *Computers & Operations Research*, *39*(9), 2033–2050. https://doi.org/10.1016/j.cor.2011.09.021

Cordeau, Jean-Fran�ois, Gendreau, M., & Laporte, G. (1997). A tabu search heuristic for periodic and multi-depot vehicle routing problems. *Networks*, *30*(2), 105–119. https://doi.org/10.1002/(SICI)1097-0037(199709)30:2<105::AID-NET5>3.0.CO;2-G

Côté, J.-F., & Potvin, J.-Y. (2009). A tabu search heuristic for the vehicle routing problem with private fleet and common carrier. *European Journal of Operational Research*, *198*(2), 464–469. https://doi.org/10.1016/j.ejor.2008.09.009

Crevier, B., Cordeau, J.-F., & Laporte, G. (2007). The multi-depot vehicle routing problem with inter-depot routes. *European Journal of Operational Research*, *176*(2). https://doi.org/10.1016/j.ejor.2005.08.015

Dantzig, G. B., & Ramser, J. H. (1959). The Truck Dispatching Problem. *Management Science*, *6*(1), 80–91. https://doi.org/10.1287/mnsc.6.1.80

Dondo, R., & Cerdá, J. (2007). A cluster-based optimization approach for the multi-depot heterogeneous fleet vehicle routing problem with time windows. *European Journal of Operational Research*, *176*(3), 1478–1507. https://doi.org/10.1016/j.ejor.2004.07.077

Dueck, G., & Scheuer, T. (1990). Threshold accepting: A general purpose optimization algorithm appearing superior to simulated annealing. *Journal of Computational Physics*, *90*(1), 161–175. https://doi.org/10.1016/0021-9991(90)90201-B

Escobar, J. W., Linfati, R., Toth, P., & Baldoquin, M. G. (2014). A hybrid Granular Tabu Search algorithm for the Multi-Depot Vehicle Routing Problem. *Journal of Heuristics*, *20*(5), 483–509. https://doi.org/10.1007/s10732-014-9247-0

Fisher, M. L., & Jaikumar, R. (1981). A generalized assignment heuristic for vehicle routing. *Networks*, *11*(2). https://doi.org/10.1002/net.3230110205

Gallardo San Salvador, J. Á. (1994). *Métodos Jerárquicos de Análisis Multivariante*. Retrieved from http://www.ugr.es/~gallardo/pdf/cluster-3.pdf

Gallego Rendón, R. A., Toro Ocampo, E. M., & Escobar Zuluaga, A. H. (2015). *Técnicas Heurísticas y Metaheurísticas.* Universidad Tecnológica de Pereira. Vicerrectoría de Investigaciones, Innovación y

Extensión. Ingenierías Eléctrica, Electrónica, Física y Ciencias de la Computación.

Gillett, B. E., & Johnson, J. G. (1976). Multi-terminal vehicle-dispatch algorithm. *Omega*, *4*(6), 711–718. https://doi.org/http://dx.doi.org/10.1016/0305-0483(76)90097-9

Giosa, I. D., Tansini, I. L., & Viera, I. O. (2002). New assignment algorithms for the multi-depot vehicle routing problem. *Journal of the Operational Research Society*, *53*(9), 977–984. https://doi.org/10.1057/palgrave.jors.2601426

Golden, B L, Magnanti, T. L., & Nguyen, H. Q. (1977). Implementing vehicle routing algorithms. *Networks*, *7*(2), 113–148. https://doi.org/10.1002/net.3230070203

Golden, Bruce L, & Wasil, E. A. (1987). Computerized Vehicle Routing in the Soft Drink Industry. *Operations Research*, *35*(1), 6–17. https://doi.org/10.1287/opre.35.1.6

Gutiérrez, R., González, A., Torres, F., & Gallardo, J. A. (1994). Métodos jerárquicos de análisis cluster. *Técnicas de Análisis de Datos Multivariable. Tratamiento Computacional*.

Ho, W., Ho, G. T. S., Ji, P., & Lau, H. C. W. (2008). A hybrid genetic algorithm for the multi-depot vehicle routing problem. *Engineering Applications of Artificial Intelligence*, *21*(4), 548–557. https://doi.org/10.1016/j.engappai.2007.06.001

Irnich, S. (2000). Multi-depot pickup and delivery problem with a single hub and heterogeneous vehicles. *European Journal of Operational Research*, *122*(2), 310–328. https://doi.org/10.1016/S0377-2217(99)00235-0

Klincewicz, J. G., Luss, H., & Pilcher, M. G. (1990). Fleet Size Planning when Outside Carrier Services Are Available. *Transportation Science*, *24*(3), 169–182. https://doi.org/10.1287/trsc.24.3.169

Klots, B., Gal, S., & Harpaz, A. (1992). Multi-depot and multi product delivery optimization problem with time and service constraints. *Israel Technical Representation*.

Kratica, J., Kostic, T., Tosic, D., Dugosija, D., & Filipovic, V. (2012). A genetic algorithm for the routing and carrier selection problem. *Computer Science and Information Systems*, *9*(1), 49–62. https://doi.org/10.2298/CSIS100425067K

Kuo, Y., & Wang, C. C. (2012). A variable neighborhood search for the multi-depot vehicle routing problem with loading cost. *Expert Systems with Applications*, *39*(8), 6949–6954. https://doi.org/10.1016/j.eswa.2012.01.024

Lalla-Ruiz, E., Expósito-Izquierdo, C., Taheripour, S., & Voß, S. (2016). An improved formulation for the multi-depot open vehicle routing problem. *OR Spectrum*, *38*(1), 175–187. https://doi.org/10.1007/s00291-015-0408-9

Laporte, G., Nobert, Y., & Taillefer, S. (1988). Solving a Family of Multi-Depot Vehicle Routing and Location-Routing Problems. *Transportation Science*, *22*(3), 161–172. https://doi.org/10.1287/trsc.22.3.161

Lavorato, M., Franco, J. F., Rider, M. J., & Romero, R. (2012, February). Imposing radiality constraints in distribution system optimization problems. *IEEE Transactions on Power Systems*, Vol. 27, pp. 172–180. https://doi.org/10.1109/TPWRS.2011.2161349

Lim, A., & Wang, F. (2004). Multi-depot vehicle routing problem: a one-stage approach. *IEEE Transactionson Automation Science and Engineering*, *4*, 397–402.

Liu, C.-Y. (2013). An Improved Adaptive Genetic Algorithm for the Multi-depot Vehicle Routing Problem with Time Window. *Journal of Networks*, *8*(5), 1035–1042. https://doi.org/10.4304/jnw.8.5.1035-1042

Liu, C., & Yu, J. (2013). Multiple depots vehicle routing based on the ant colony with the genetic algorithm. *Journal of Industrial Engineering and Management*, *6*(4), 1013–1026. https://doi.org/10.3926/jiem.747

Liu, R., & Jiang, Z. (2012). The close–open mixed vehicle routing problem. *European Journal of Operational Research*, *220*(2), 349–360. https://doi.org/10.1016/j.ejor.2012.01.061

Liu, R., Jiang, Z., & Geng, N. (2014). A hybrid genetic algorithm for the multi-depot open vehicle routing problem. *OR Spectrum*, *36*(2), 401–421. https://doi.org/10.1007/s00291-012-0289-0

Min, H., Current, J., & Schilling, D. (2003). The multiple depot vehicle routing problem with backhauling. *Journal of Business Logistics*, *13*(1), 259–288.

Mirabi, M., Fatemi Ghomi, S. M. T., & Jolai, F. (2010). Efficient stochastic hybrid heuristics for the multi-depot vehicle routing problem. *Robotics and Computer-Integrated Manufacturing*, *26*(6), 564–569. https://doi.org/10.1016/j.rcim.2010.06.023

Mirledy, E., Ocampo, T., & Echeverri, M. G. (2016). *Solución del problema de localización y ruteo usando un modelo matemático exible y considerando efectos ambientales*.

Montoya-Torres, J. R., López Franco, J., Nieto Isaza, S., Felizzola Jiménez, H., & Herazo-Padilla, N. (2015). A literature review on the vehicle routing problem with multiple depots. *Computers & Industrial Engineering*, *79*, 115–129. https://doi.org/10.1016/j.cie.2014.10.029

Nagy, G., & Salhi, S. (2005a). Heuristic algorithms for single and multiple depot vehicle routing problems with pickups and deliveries. *European Journal of Operational Research*, *162*(1), 126–141. https://doi.org/10.1016/j.ejor.2002.11.003

Nagy, G., & Salhi, S. (2005b). Heuristic algorithms for single and multiple depot vehicle routing problems with pickups and deliveries. *European Journal of Operational Research*, *162*(1), 126–141. https://doi.org/10.1016/j.ejor.2002.11.003

Ocampo, E. M. T., Castaño, A. H. D., & Zuluaga, A. H. E. (2016). Desempeño de las técnicas de agrupamiento para resolver el problema de ruteo con múltiples depósitos. *Tecno Lógicas*, *19*(36), 49–62.

Pepin, A.-S., Desaulniers, G., Hertz, A., & Huisman, D. (2008). A comparison of five heuristics for the multiple depot vehicle scheduling problem. *Journal of Scheduling*, *12*(1), 17. https://doi.org/10.1007/s10951-008-0072-x

Perl, J. (1987). The Multi-Depot Routing Allocation Problem. *American Journal of Mathematical and Management Sciences*, *7*(1–2), 7–34. https://doi.org/10.1080/01966324.1987.10737206

Perl, J., & Daskin, M. S. (1985). A warehouse location-routing problem. *Transportation Research Part B: Methodological*, *19*(5), 381–396. https://doi.org/http://dx.doi.org/10.1016/0191-2615(85)90052-9

Pichka, K., Bajgiran, A. H., Petering, M. E. H., Jang, J., & Yue, X. (2018). The two echelon open location routing problem: Mathematical model and hybrid heuristic. *Computers and Industrial Engineering*, *121*, 97–112. https://doi.org/10.1016/j.cie.2018.05.010

Pisinger, D., & Ropke, S. (2007). A general heuristic for vehicle routing problems. *Computers and Operations Research*, *34*(8), 2403–2435. https://doi.org/10.1016/j.cor.2005.09.012

Polacek, M., Hartl, R. F., Doerner, K., & Reimann, M. (2004). A Variable Neighborhood Search for the Multi Depot Vehicle Routing Problem with Time Windows. *Journal of Heuristics*, *10*(6), 613–627. https://doi.org/10.1007/s10732-005-5432-5

Prodhon, C., & Prins, C. (2014). A survey of recent research on location-routing problems. *European Journal of Operational Research*, *238*(1), 1–17. https://doi.org/10.1016/j.ejor.2014.01.005

Raff, S. (1983). Routing and scheduling of vehicles and crews. *Computers & Operations Research*, *10*(2), 63–211. https://doi.org/10.1016/0305-0548(83)90030-8

Renaud, J., Laporte, G., & Boctor, F. F. (1996). A tabu search heuristic for the multi-depot vehicle routing problem. *Computers & Operations Research*, *23*(3), 229–235. https://doi.org/10.1016/0305-0548(95)O0026-P

Salhi, S., & Sari, M. (1997). A multi-level composite heuristic for the multi-depot vehicle fleet mix problem. *European Journal of Operational Research*, *103*(1), 95–112. https://doi.org/10.1016/S0377-2217(96)00253-6

Sariklis, D., & Powell, S. (2000). A heuristic method for the open vehicle routing problem. *Journal of the Operational Research Society*, *51*(5), 564–573. https://doi.org/10.1057/palgrave.jors.2600924

Stenger, A., Vigo, D., Enz, S., & Schwind, M. (2013). An Adaptive Variable Neighborhood Search Algorithm for a Vehicle Routing Problem Arising in Small Package Shipping. *Transportation Science*, *47*(1), 64–80. https://doi.org/10.1287/trsc.1110.0396

Subramanian, A. (2012). *Heuristic, Exact and Hybrid Approaches for Vehicle Routing Problems*.

Subramanian, A., Uchoa, E., & Ochi, L. S. (2013). A hybrid algorithm for a class of vehicle routing problems. *Computers and Operations Research*, *40*(10), 2519–2531. https://doi.org/10.1016/j.cor.2013.01.013

Surekha, P., & Sumathi, S. (2011). Solution To Multi-Depot Vehicle Routing Problem Using Genetic Algorithms. *World Applied Programming*, *1*(3), 118–131. Retrieved from http://www.waprogramming.com/index.php?action=journal&page=showpaper&jid=1&iid=3&pid=12

Tarantilis, C. D., Ioannou, G., Kiranoudis, C. T., & Prastacos, G. P. (2005). Solving the Open Vehicle Routeing Problem Via a Single Parameter Metaheuristic Algorithm. *The Journal of the Operational Research Society*, *56*(5), 588–596. Retrieved from http://www.jstor.org/stable/4102111

Tarantilis, C. D., & Kiranoudis, C. T. (2002). Distribution of fresh meat. *Journal of Food Engineering*, *51*(1), 85–91. https://doi.org/10.1016/S0260-8774(01)00040-1

Thangiah, S. R., & Salhi, S. (2001). Genetic clustering: An adaptive heuristic for the multidepot vehicle routing problem. *Applied Artificial Intelligence*, *15*(4), 361–383. https://doi.org/10.1080/08839510151087293

Tillman, F. A. (1969). The Multiple Terminal Delivery Problem with Probabilistic Demands. *Transportation Science*, *3*(3), 192–204. https://doi.org/10.1287/trsc.3.3.192

Tillman, F. A., & Cain, T. M. (1972). An Upperbound Algorithm for the Single and Multiple Terminal

Delivery Problem. *Management Science*, *18*(11), 664–682. https://doi.org/10.1287/mnsc.18.11.664

Tillman, F. A., & Hering, R. W. (1971). A study of a look-ahead procedure for solving the multiterminal delivery problem. *Transportation Research*, *5*(3), 225–229. https://doi.org/10.1016/0041-1647(71)90023-2

Toro-ocampo, E., Gallego-rendón, R., & Franco-baquero, J. F. (2016). *Modelo matemático para resolver el problema de localización y ruteo con restricciones de capacidad considerando flota propia y subcontratada Mathematical Model for Capacitated Location Routing Problem with Private Fleet and Common Carrier*. (número 3), 357–369.

Toro-Ocampo, E. M., Franco-Baquero, J. F., & Gallego-Rendón, R. A. (2016). Modelo matemático para resolver el problema de localización y ruteo con restricciones de capacidad considerando flota propia y subcontratada. *Ingeniería, Investigación y Tecnología*, *17*(3), 357–369. https://doi.org/10.1016/j.riit.2016.07.006

Toro, E. M., Domínguez, A. H., & Escobar, A. H. (2013). Performance of clustering techniques for solving multi depot vehicle routing problem. *Tecno Lógicas*, *19*(36), 289–301. Retrieved from http://www.scielo.org.co/scielo.php?script=sci_arttext&pid=S0123-77992016000100005

Toro O., E. M., Escobar Z., A. H., & Granada E., M. (2015). LITERATURE REVIEW ON THE VEHICLE ROUTING PROBLEM IN THE GREEN TRANSPORTATION CONTEXT. *Luna Azul*, (42), 362–387. https://doi.org/10.17151/luaz.2016.42.21

Venkata Narasimha, K., Kivelevitch, E., Sharma, B., & Kumar, M. (2013). An ant colony optimization technique for solving min-max Multi-Depot Vehicle Routing Problem. *Swarm and Evolutionary Computation*, *13*, 63–73. https://doi.org/10.1016/j.swevo.2013.05.005

Vidal, T., Crainic, T. G., Gendreau, M., Lahrichi, N., & Rei, W. (2012). A Hybrid Genetic Algorithm for Multidepot and Periodic Vehicle Routing Problems. *Operations Research*, *60*(3), 611–624. https://doi.org/10.1287/opre.1120.1048

Vidal, T., Crainic, T. G., Gendreau, M., & Prins, C. (2014). Implicit depot assignments and rotations in vehicle routing heuristics. *European Journal of Operational Research*, *237*(1), 15–28. https://doi.org/10.1016/j.ejor.2013.12.044

Wang, H., Du, L., & Ma, S. (2014). Multi-objective open location-routing model with split delivery for optimized relief distribution in post-earthquake. *Transportation Research Part E: Logistics and Transportation Review*, *69*, 160–179. https://doi.org/10.1016/j.tre.2014.06.006

Wassan, N. A., & Osman, I. H. (2002). Tabu search variants for the mix fleet vehicle routing problem. *Journal of the Operational Research Society*, *53*(7), 768–782. https://doi.org/10.1057/palgrave.jors.2601344

Wren, A., & Holliday, A. (1972). Computer Scheduling of Vehicles from One or More Depots to a Number of Delivery Points. *Journal of the Operational Research Society*, *23*(3), 333–344. https://doi.org/10.1057/jors.1972.53

Xu, Y., Wang, L., & Yang, Y. (2012). A New Variable Neighborhood Search Algorithm for the Multi Depot Heterogeneous Vehicle Routing Problem with Time Windows. *Electronic Notes in Discrete Mathematics*, *39*, 289–296. https://doi.org/10.1016/j.endm.2012.10.038

Yao, B., Hu, P., Zhang, M., & Tian, X. (2014). Improved Ant Colony Optimization for Seafood Product

Delivery Routing Problem. *PROMET - Traffic&Transportation*, *26*(1). https://doi.org/10.7307/ptt.v26i1.1478

Yu, B., Yang, Z.-Z., & Xie, J.-X. (2011). A parallel improved ant colony optimization for multi-depot vehicle routing problem. *Journal of the Operational Research Society*, *62*(1), 183–188. https://doi.org/10.1057/jors.2009.161

ZHANG, J. U. N., TANG, J., & FUNG, R. Y. K. (2011). A SCATTER SEARCH FOR MULTI-DEPOT VEHICLE ROUTING PROBLEM WITH WEIGHT-RELATED COST. *Asia-Pacific Journal of Operational Research*, *28*(03), 323–348. https://doi.org/10.1142/S0217595911003260